03.10.03

日本国特許庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出願年月日
Date of Application: ２００２年１０月　３日

出願番号
Application Number: 特願２００２－２９１７０８

[ST. 10/C]: [ＪＰ２００２－２９１７０８]

出願人
Applicant(s): 株式会社インフォーエス

２００３年１２月１８日

特許庁長官
Commissioner,
Japan Patent Office

今井康夫

【書類名】　特許願
【整理番号】　020320P531
【あて先】　特許庁長官殿
【国際特許分類】　H04L 12/00
【発明者】
　【住所又は居所】　東京都多摩市鶴牧６丁目１６番地４－５０２
　【氏名】　佐藤　哲朗
【発明者】
　【住所又は居所】　東京都練馬区上石神井２丁目６番１２号
　【氏名】　河口　文法
【特許出願人】
　【識別番号】　502254981
　【氏名又は名称】　有限会社ディシーエル
【代理人】
　【識別番号】　100102934
　【弁理士】
　【氏名又は名称】　今井　彰
【手数料の表示】
　【予納台帳番号】　050728
　【納付金額】　21,000円
【提出物件の目録】
　【物件名】　明細書　1
　【物件名】　図面　1
　【物件名】　要約書　1
【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　ビットストリングの照合方法

【特許請求の範囲】

【請求項１】　被検索ビットストリングを複数ビットの部分被検索ビットストリングに分けて、予め登録された複数の登録ビットパターンと多段階で照合するビットストリングの照合方法であって、１つの照合段階は、

前記被検索ビットストリングから、現段階の前記部分被検索ビットストリングを選択し、その部分被検索ビットストリングが取り得る全ての値と照合する全照合工程と、

前記登録ビットパターンの部分ビットパターンであって、先行する段階から得られる照合継続情報により決まる、前記部分被検索ビットストリングに対応する段階の部分登録ビットパターンを示す前記現段階のマスクデータを、前記全照合工程と前後して、または並列に、メモリよりロードするマスクロード工程と、

前記全照合工程の結果と前記現段階のマスクデータより、前記部分被検索ビットストリングに一致する前記部分登録ビットパターンの有無を少なくとも示す照合結果を得る判定工程と、

前記照合結果により、前記現段階のマスクデータに対応して記録された前記部分登録ビットパターンのそれぞれに続く次の段階のマスクデータのアドレスが示されたアドレステーブルの中から、前記一致する部分登録ビットパターンに続く次の段階のマスクデータのアドレスを含む前記照合継続情報を出力する出力工程とを有するビットストリングの照合方法。

【請求項２】　請求項１において、前記全照合工程とマスクロード工程とを並列に行う、ビットストリングの照合方法。

【請求項３】　請求項１において、前記判定工程では、前記部分被検索ビットストリングに値が最も近い最大または最小の前記登録部分ビットパターンの有無を示す照合結果が得られ、

前記出力工程では、前記一致する部分登録ビットパターンがないときに、前記アドレステーブルの中から、前記最大または最小の部分登録ビットパターンに続く次の段階のマスクデータのアドレスを含む前記照合継続情報を出力する、ビッ

トストリングの照合方法。

【請求項４】　請求項３において、前記一致する部分登録ビットパターンがあり、前記最大または最小の部分登録ビットパターンがあるときに、前記最大または最小の部分登録ビットパターンに続く次の段階のマスクデータのアドレスを候補アドレスとして記憶する候補アドレス記憶工程を有し、

前記出力工程では、前記一致する部分登録ビットパターンおよび前記最大または最小の部分登録ビットパターンがないときに、前記候補アドレスを含む前記照合継続情報を出力する、ビットストリングの照合方法。

【請求項５】　請求項３において、前記最大または最小の部分登録ビットパターンに続く前記現段階のマスクデータのアドレスを含む前記照合継続情報が与えられたときは、前記全照合工程および判定工程をバイパスする工程を有し、

前記出力工程では、前記現段階のマスクデータに対応する前記アドレステーブルから、前記現段階のマスクデータに示された最大または最小の部分登録ビットパターンに続く次の段階のマスクデータのアドレスを含む前記照合継続情報を出力する、ビットストリングの照合方法。

【請求項６】　請求項３において、前記マスクデータのそれぞれに対して、そのマスクデータに示された最大また最小の部分登録ビットパターンに対応して決まる最大または最小の前記登録ビットパターンを示すバイパスデータが記憶されており、

前記最大または最小の部分登録ビットパターンに続く前記現段階のマスクデータのアドレスを含む前記照合継続情報が与えられたときは、前記全照合工程、マスクロード工程および判定工程をバイパスする工程と有し、

前記出力工程では、前記現段階のマスクデータに対応する前記バイパスデータを含む最終照合情報を出力して前記被検索ビットストリングの照合を終了する、ビットストリングの照合方法。

【請求項７】　請求項１において、前記マスクデータおよびアドレステーブルを更新することにより、前記登録ビットパターンを追加または削除する更新工程を有する、ビットストリングの照合方法。

【請求項８】　請求項１において、複数の前記照合段階がパイプライン方式

で実行される、ビットストリングの照合方法。

【請求項９】　請求項１において、複数の前記照合段階の間では、前記照合継続情報を含むデータがパケット化されて伝達される、ビットストリングの照合方法。

【請求項１０】　請求項３において、複数の前記照合段階を備えた分類過程を複数有し、前記最大または最小の部分登録ビットパターンから導かれる最大または最小の前記登録ビットパターンは複数の分類結果を含む範囲を示しており、

前記分類過程により得られる前記複数の分類結果の論理積を演算し、最終の分類結果を得る論理演算工程を有する、ビットストリングの照合方法。

【請求項１１】　請求項１０において、前記論理演算工程では、前記複数の分類結果の論理積を行列演算し、最終の分類結果を得る、ビットストリングの照合方法。

【請求項１２】　請求項１に記載の照合方法により、前記被検索ビットストリームを照合する工程と、

その結果に基づき前記被検索ビットストリングを備えたデータを管理する工程とを有するデータ管理方法。

【請求項１３】　請求項１０に記載の照合方法により、ＩＰアドレス、ポート番号およびプロトコルの少なくともいずれかを含むパケットを管理するためのビットストリングを前記被検索ビットストリングとし、それが属する前記範囲を照合する工程と、

その結果に基づき前記被検索ビットストリングを備えた前記パケットを管理する工程とを有するデータ管理方法。

【請求項１４】　被検索ビットストリングを複数ビットの部分被検索ビットストリングに分けて、予め登録された複数の登録ビットパターンと多段階で照合するための１つの照合段階を実行する照合装置であって、

前記被検索ビットストリングから、現段階の前記部分被検索ビットストリングを選択し、その部分被検索ビットストリングが取り得る全ての値と照合する全照合手段と、

前記登録ビットパターンの部分ビットパターンであって、先行する段階から得

られる照合継続情報により決まる、前記部分被検索ビットストリングに対応する段階の部分登録ビットパターンを示す前記現段階のマスクデータをメモリよりロードするマスクロード手段と、

前記全照合手段の結果と前記現段階のマスクデータより、前記部分被検索ビットストリングに一致する前記部分登録ビットパターンの有無を少なくとも示す照合結果を出力する判定手段と、

前記照合結果により、前記現段階のマスクデータに対応して記録された前記部分登録ビットパターンのそれぞれに続く次の段階のマスクデータのアドレスが示されたアドレステーブルの中から、前記一致する部分登録ビットパターンに続く次の段階のマスクデータのアドレスを含む前記照合継続情報を出力する出力手段とを有する照合装置。

【請求項１５】　請求項１４において、前記判定手段は、前記部分被検索ビットストリングに値が最も近い最大または最小の前記登録部分ビットパターンの有無を示す照合結果を出力し、

前記出力手段は、前記一致する部分登録ビットパターンがないときに、前記アドレステーブルの中から、前記最大または最小の部分登録ビットパターンに続く次の段階のマスクデータのアドレスを含む前記照合継続情報を出力する、照合装置。

【請求項１６】　請求項１５において、前記出力手段は、前記一致する部分登録ビットパターンがあり、前記最大または最小の部分登録ビットパターンがあるときに、前記最大または最小の部分登録ビットパターンに続く次の段階のマスクデータのアドレスを候補アドレスとして出力し、さらに、

前記一致する部分登録ビットパターンおよび前記最大または最小の部分登録ビットパターンがないときに、前記候補アドレスを含む前記照合継続情報を出力する、照合装置。

【請求項１７】　請求項１５において、前記出力手段は、前記最大または最小の部分登録ビットパターンに続く前記現段階のマスクデータのアドレスを含む前記照合継続情報が与えられたときは、前記現段階のマスクデータに対応する前記アドレステーブルから、前記現段階のマスクデータに示された最大または最小

の部分登録ビットパターンに続く次の段階のマスクデータのアドレスを含む前記照合継続情報を出力する、照合装置。

【請求項１８】　請求項１５において、前記マスクデータのそれぞれに対して、そのマスクデータに示された最大また最小の部分登録ビットパターンに対応して決まる最大または最小の前記登録ビットパターンを示すバイパスデータが記憶されており、

前記出力手段は、前記最大または最小の部分登録ビットパターンに続く前記現段階のマスクデータのアドレスを含む前記照合継続情報が与えられたときは、前記現段階のマスクデータに対応する前記バイパスデータを最終照合情報として出力する、照合装置。

【請求項１９】　請求項１４において、前記マスクデータおよびアドレステーブルを更新することにより、前記登録ビットパターンを追加または削除する更新手段を有する、照合装置。

【請求項２０】　請求項１４に記載の照合装置を複数有する分類装置。

【請求項２１】　請求項２０において、複数の前記照合装置はパイプラインで接続されている、分類装置。

【請求項２２】　請求項２０において、複数の前記照合装置の間では、前記照合継続情報を含むデータがパケット化されて伝達される、分類装置。

【請求項２３】　請求項２０において、前記メモリに対してデータを入出力するキャッシュ装置を有する、分類装置。

【請求項２４】　請求項２３において、前記メモリには、前記マスクデータおよび前記アドレステーブルを含む照合テーブルが記憶されており、前記キャッシュ装置のキャッシュラインサイズが前記照合テーブルのサイズと同一である、分類装置。

【請求項２５】　請求項２４において、最初の照合テーブルを指定する検索種別情報を受信する手段を有し、

前記キャッシュ装置は、それぞれの検索種別および前記照合段階の単位でキャッシュラインを割り当て、前記検索種別および照合段階の単位で前記キャッシュラインを管理する管理手段を備えている、分類装置。

【請求項２６】　請求項２４において、最初の照合テーブルを指定する検索種別情報を受信する手段を有し、

前記メモリには、それぞれの検索種別および前記照合段階の単位で割り当てられた個別アドレス領域に前記照合テーブルが記憶されており、

前記キャッシュ装置のキャッシュラインは、前記個別アドレス領域単位で割り当てられている、分類装置。

【請求項２７】　請求項２３において、前記キャッシュ装置は、照合テーブルがオンキャッシュでなければ前記照合装置にその旨を通知し、その照合装置の処理を中断して待ち行列に戻す、分類装置。

【請求項２８】　請求項２０において、当該分類装置の分類結果を記憶した履歴キャッシュ装置であって、前記被検索ビットストリングを前記照合装置に供給する前に、当該分類装置の分類結果と前記被検索ビットストリングとを比較する履歴キャッシュ装置を有する分類装置。

【請求項２９】　請求項２０に記載の分類装置を有し、前記最大または最小の部分登録ビットパターンから導かれる最大または最小の前記登録ビットパターンは複数の分類結果を含む範囲を示しており、

さらに、前記分類装置により得られる複数の分類結果の論理積を演算し、最終の分類結果を得る論理演算装置を有する検索装置。

【請求項３０】　請求項２９において、複数の前記分類装置を有する検索装置。

【請求項３１】　請求項２９において、前記論理演算装置は、前記複数の分類結果の論理積を行列演算し、最終の分類結果を得る、検索装置。

【請求項３２】　請求項２９において、前記メモリに対してデータを入出力するキャッシュ装置を有する、検索装置。

【請求項３３】　請求項２９において、当該検索装置の検索結果を記憶した履歴キャッシュ装置であって、前記被検索ビットストリングを前記分類装置に供給する前に、当該検索装置の検索結果と前記被検索ビットストリングとを比較する履歴キャッシュを有する検索装置。

【請求項３４】　請求項２０に記載の分類装置と、

前記分類装置の出力に基づき前記被検索ビットストリングを備えたデータを管理する手段とを有するデータ管理装置。

【請求項３５】　請求項２９に記載の検索装置を有し、

前記検索装置に、ＩＰアドレス、ポート番号およびプロトコルの少なくともいずれかを含むパケットを管理するためのビットストリングを前記被検索ビットストリングとして供給する手段と、

前記検索装置の出力に基づき前記被検索ビットストリングを備えた前記パケットを管理する手段とを有するデータ管理装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、ＩＰアドレスなどのビットストリングを検索あるいは照合に適した装置および方法に関するものである。

【０００２】

【従来の技術】

インターネットなどのコンピュータネットワークを介してデータを送受信する際には所定のプロトコルが適用され、インターネットのネットワーク層におけるプロトコルはインターネットプロトコル（ＩＰ）と呼ばれている。このＩＰは情報をパケット化して所定の目的地まで移動させるが、そのために、情報を格納したペイロードにＩＰヘッダをつけたパケットを送受信する。そして、ＩＰヘッダの情報に基づいて、各々のパケットは所定のルートに配送され、また、受信先では、ＩＰヘッダの情報に基づいてペイロードの情報が送信元の状態となるように組み立てられる。

【０００３】

また、ＩＰヘッダの情報は、ルータおよびファイアウォールなどのネットワークを管理する装置あるいはソフトウェア（プログラムあるいはプログラム製品）で参照され、ルーティング、パケットフィルタリング、ＳＰＤ（セキュリティ・ポリシーデータベース）、ポリシールーティング、ＱｏＳなどの様々な目的のために照合され、パケットの送受信あるいは転送については常に利用される。

【０００４】

さらに、今後、モバイルＩＰが実現されると、ルーティングテーブルは動的に変動する可能性があり、また、現状のＩＰｖ４（バージョン４）では、受信先アドレス（ＳＩＰ）および送信元アドレス（ＤＩＰ）は３２ビットであるのに対し、ＩＰｖ６（バージョン６）では１２８ビットに拡張される。したがって、今後、ＩＰアドレスの検索における処理速度をさらに向上させることが、インターネットなどのコンピュータネットワークを用いてデータ交換をする上で非常に重要な技術となる。

【０００５】

たとえば、特開平１１－８８４２７号公報には、ソートされた複数のエントリからなるルーティングテーブルに対して検索ＩＰアドレスを大小比較しながら一致するＩＰアドレスを検索する技術が開示されているが、この方法によりルーティングあるいはその他の目的のために１２８ビットのＩＰアドレスを複数含むビットストリングを検索しようとすると膨大な時間が必要になり、実用的ではない。

【０００６】

【特許文献１】

特開平１１－８８４２７号公報

【０００７】

【発明が解決しようとする課題】

今後、ＩＰアドレスに限らず、検索対象となるビットストリング（ビットストリームあるいはビットパターン）の情報量は増大する傾向にあり、光通信が本格化する場合は、Ｇビットクラスのビットストリングの検索機能が要求される。膨大なデータ量のビットストリングの検索方法は幾つか提案されている。まず、ソフトウェア方式は、検索機能は十分だが、速度的には不十分である。

【０００８】

ハードウェアを用いた方式では、ＣＡＭ（Content-Addressable Memory）方式がある。しかしながら、１個のＣＡＭを使った繰り返し検索での最長一致検索、あるいは複数個のＣＡＭを並列に使っての最長一致検索を実現するところまでし

かできず、パケットフィルタリングやステートフルインスペクションに要求される多次元範囲、たとえば、ソースＩＰ・デスティネーションＩＰ・ソースポート・デスティネーションポート・プロトコルを含む範囲の一致検索を実現するのは非常に困難である。また、最長一致検索の対象としてＩＰルーティングを考えた場合、ＣＩＤＲ形式での可変プレフィックス長への対応は、繰り返し型の繰り返し回数増や多数個ＣＡＭ並列型でおこなうことになり、前者であればスループットやレイテンシ面での性能不足、後者であれば経路エントリの分布ばらつきにより、ＣＡＭエントリ利用率が悪くなるという問題点を抱える。

【０００９】

登録エントリ数増に対しては複数ＣＡＭチップのカスケード接続でしか対応できない。被検索キーのビット長の増加に対しては、チップ自体のスペック変更でしか対応できず、全エントリのＣＡＭメモリビット増が必要である。多チップ構成での対応の可能性については、ＣＡＭは多エントリ並列に照合されるため、複数チップ間で連動した照合をおこなうためには各エントリの照合状態をチップ間で継承しなければならず、ＣＡＭに求められる大量エントリ数を考えると現実的ではない。

【００１０】

部分的にＣＡＭを利用し、ＣＡＭにより検索対象エントリを絞り込み、その後パトリシアツリー等をソフトウェアにより検索するという方式もある。この方式ではツリー検索にソフトウェアを用いるため、完全ソフトウェア方式と同様の問題がある。また、ツリー検索部をハードウェア化した場合でも、例えばツリー構造がパトリシアであれば登録削除に複雑なツリー再構成が伴い、上位アプリケーションソフトウェアに過度なテーブル管理コストを強いることになる。

【００１１】

マスキングが可能で、階層型の検索を可能としたターナリーＣＡＭ方式がある。この場合、ハードウェアそのものが提供する検索機能は、完全一致（ＣＡＭ）と最長一致である。ＴＣＡＭは原理的にはエントリのＤＣ(ドントケア)ビットをも一致と見なす完全一致であり、照合部からは複数の一致エントリが出力される。そのため、照合部の後段にプレフィックス長あるいはエントリＮｏ等による何

らかのプライオリティエンコーダがないと、「複数ヒットから最長一致」の絞り込みをおこなえない。

【００１２】

従って、ＴＣＡＭにおいてはエントリの登録削除についての管理が煩雑となる。特にエントリＮｏがプライオリティとなる方式のＴＣＡＭの場合、エントリの追加にともない、既存エントリの再配置を上位アプリケーションソフトウェア側が自発的に行わねばならない。この既存エントリの再配置はプレフィックス長をキーとしたソートであり、複数回のエントリ位置交換が要求される。プレフィックス長だけが問題なので通常のソートよりは低コストであるが、ソートそのものは排除できない。すなわち、ＴＣＡＭで範囲一致は実現できるが、それはエントリ管理をおこなう上位アプリケーションソフトウェアによるものであり、範囲表現を複数の最長一致表現へと分割し、最長一致へ還元して実現することになる。そのため、範囲一致のルール管理（削除・登録）が煩雑となる。つまり、ＴＣＡＭにおける範囲一致は、静的なエントリからなる検索については実用性があったとしても、動的に変化するエントリからなる検索については、かなり上位アプリケーションソフトウェアへの負荷が高いといえる。あるいは、これら複雑なテーブル管理をおこなう専用のコプロセッサをＴＣＡＭ自体とは別途用意することになる。

【００１３】

以上のことから、範囲一致は無論、最長一致であっても、登録エントリがダイナミックに追加削除されていく性質のある検索にはＴＣＡＭは向かないといえる。登録エントリ数増に対しては複数のＴＣＡＭチップのカスケード接続でしか対応できないことはＣＡＭと同様である。被検索キーのビット長の増加に対しても、チップ自体のスペック変更でしか対応できず、全エントリのＴＣＡＭメモリビット増が必要である。多チップ構成での対応の可能性については、ＴＣＡＭは多エントリ並列に照合されるため、複数チップ間で連動した照合をおこなうためには各エントリの照合状態をチップ間で継承しなければならず、ＴＣＡＭに求められる大量エントリ数を考えると現実的ではない。

【００１４】

そこで、本発明においては、範囲一致を始め、完全一致および最長一致といった検索機能を提供可能であり、パケットフィルタリングやステートフルインスペクションにおいても上位アプリケーションソフトウェアの負担が軽いビットストリングの照合方法および照合装置を提供することを目的としている。特に、エントリの追加削除にともなうエントリ移動による、上位アプリケーションによるテーブル管理のオーバヘッドを縮小または排除できるビットストリングの照合方法および照合装置を提供することを目的としている。

【００１５】

さらに、メモリとして、ＳＤＲＡＭ等通常のＤＲＡＭ（あるいはＳＲＡＭ）を使用可能とし、十分な検索速度を提供しながら、ＣＡＭ系の専用メモリよりもはるかに廉価で大容量を実現できる照合方法および装置を提供することも本発明の目的である。そして、そのような照合装置は、収容可能なエントリ数はＳＤＲＡＭ容量にのみ規定されるため、スケーラブル性が高い。また、メモリ管理が柔軟に行えるため、複数種の検索を単一のハードウェアセットで柔軟におこなえるものになる。また、検索対象となる被検索ビットストリング（キー）のビット長増に対してもハードウェア自体のアルゴリズム的な変更は必要なく、ユニットを追加したり、一連の処理を繰り返すことで対応可能な照合方法および照合装置を提供することも目的としている。

【００１６】

【課題を解決するための手段】

本発明のビットストリングの照合方法においては、被検索ビットストリングを複数ビットの部分被検索ビットストリングに分けて、予め登録された複数の登録ビットパターンと多段階で照合する方法を採用する。そして、１つの照合段階は、被検索ビットストリングから、現段階の部分被検索ビットストリングを選択し、その部分被検索ビットストリングが取り得る全ての値と照合する全照合工程と、登録ビットパターンの部分ビットパターンであって、先行する段階から得られる照合継続情報により決まる、部分被検索ビットストリングに対応する段階の部分登録ビットパターンを示す現段階のマスクデータを、全照合工程と前後して、または並列に、メモリよりロードするマスクロード工程と、全照合工程の結果と

現段階のマスクデータより、部分被検索ビットストリングに一致する部分登録ビットパターンの有無を少なくとも示す照合結果を得る判定工程と、照合結果により、現段階のマスクデータに対応して記録された部分登録ビットパターンのそれぞれに続く次の段階のマスクデータのアドレスが示されたアドレステーブルの中から、一致する部分登録ビットパターンに続く次の段階のマスクデータのアドレスを含む照合継続情報を出力する出力工程とを有する。

【００１７】

この照合方法を実現する照合装置として、本発明においては、被検索ビットストリングを複数ビットの部分被検索ビットストリングに分けて、予め登録された複数の登録ビットパターンと多段階で照合するための１つの照合段階を実行する照合装置を提供する。この照合装置は、被検索ビットストリングから、現段階の部分被検索ビットストリングを選択し、その部分被検索ビットストリングが取り得る全ての値と照合する全照合手段と、登録ビットパターンの部分ビットパターンであって、先行する段階から得られる照合継続情報により決まる、部分被検索ビットストリングに対応する段階の部分登録ビットパターンを示す現段階のマスクデータをメモリよりロードするマスクロード手段と、全照合手段の結果と現段階のマスクデータより、部分被検索ビットストリングに一致する部分登録ビットパターンの有無を少なくとも示す照合結果を出力する判定手段と、照合結果により、現段階のマスクデータに対応して記録された部分登録ビットパターンのそれぞれに続く次の段階のマスクデータのアドレスが示されたアドレステーブルの中から、一致する部分登録ビットパターンに続く次の段階のマスクデータのアドレスを含む照合継続情報を出力する出力手段とを有する。

【００１８】

これらの照合方法および照合装置は、完全一致および最長一致の検索機能を提供する。さらに、判定工程および判定手段において、部分被検索ビットストリングに値が最も近い最大または最小の登録部分ビットパターンの有無を示す照合結果が得られると、出力工程および出力手段では、一致する部分登録ビットパターンがないときに、アドレステーブルの中から、最大または最小の部分登録ビットパターンに続く次の段階のマスクデータのアドレスを含む照合継続情報を出力す

ることにより、検索機能として範囲一致を含めた照合方法および照合装置を提供できる。ここで、部分被検索ビットストリングに値が最も近い最大または最小の部分登録ビットパターンとは、部分被検索ビットストリングより値が小さくて最大、または値が大きくて最小の部分登録ビットパターンを意味する。最大の部分登録ビットパターンに続くアドレスが選択されることにより、以下で説明する右倒れの範囲検索が可能となる。また、最小の部分登録ビットパターンに続くアドレスが選択されることにより、以下で説明する左倒れの範囲検索が可能となる。したがって、右倒れ型の検索においては、以下で説明する最大の側が選択され、左倒れ型の検索においては、以下で説明する最小の側が選択される。

【００１９】

また、本発明の照合方法および照合装置においては、全照合工程および全照合手段により、部分被検索ビットストリングが取り得る全ての値と照合し、部分被検索ビットストリングに対応する段階の部分登録ビットパターン（以降ではエントリとも称する）を示す現段階のマスクデータから完全一致または範囲一致を導く。マスクデータは、部分エントリのビット列そのものである必要はなく、むしろ、部分エントリの値を直接またはビットパターンあるいはフラグなどにより部分エントリと対応するような形式で示すデータであることがデータ量の点から望ましく、予め部分被検索ビットストリングが取り得る値が決まっているので、そのようなデータでロードすることが可能である。予め部分被検索ビットストリングが取り得る全ての値と照合するので、エントリを追加する際は、基本的にはマスクデータを更新するだけで良く、エントリの追加削除にともなうエントリ移動は発生しない。したがって、上位アプリケーションによるテーブル管理のオーバヘッドを縮小または排除できる。このため、パケットフィルタリングやステートフルインスペクションにおいても上位アプリケーションソフトウェアの負担が軽いビットストリングの照合方法および照合装置を提供できる。

【００２０】

部分被検索ビットストリングが取り得る全ての値と照合する処理は、比較器やルックアップテーブルを用いてハードウェアで高速に実行可能であり、多段階に分割することによりハードウェアの規模も大きくならない。部分被検索ビットス

トリングのビット長は限定されるものではないが、ビット長が長くなればハードウェアは大きくなり、ビット長が短ければ照合段数が増加する。現状では、部分被検索ビットストリングのビット長としては４から５ビット程度が好ましいと考えられる。さらに、検索対象となる被検索ビットストリング（以降ではキーとも称する）のビット長増に対してもハードウェア自体のアルゴリズム的な変更は必要なく、ユニットを追加したり、一連の処理を繰り返すことで対応可能となる。

【００２１】

そして、全照合工程とマスクロード工程とを並列に行うことが可能であり、多段階に分割することに伴い発生するメモリからマスクデータをロードするオーバヘッドの時間も大幅に削減できる。したがって、特殊な構造で高速なメモリを使わなくても、十分な検索速度を発揮でき、廉価で大容量を実現する照合方法および装置を提供できる。そのような照合装置は、収容可能なエントリ数はＳＤＲＡＭ容量にのみ規定されるため、スケーラブル性が高く、また、メモリ管理が柔軟に行えるため、複数種の検索を単一のハードウェアセットで柔軟におこなえるものになることは上述した通りである。

【００２２】

さらに、マスクデータをロードする時間のオーバヘッドを削減するには、メモリに対してデータを入出力するキャッシュ装置を設けることが有効であり、キャッシュメモリの効率を向上し、ミスヒット率を低減するには、メモリ上の記憶単位であり、マスクデータおよびアドレステーブルを含む照合テーブルのサイズとキャッシュラインサイズが同一であることが好ましい。

【００２３】

さて、範囲一致検索においては、一致する部分登録ビットパターンがあり、最大または最小の部分登録ビットパターンがあるときには、最大または最小の部分登録ビットパターンに続く次の段階のマスクデータのアドレスを候補アドレスとして記憶あるいは出力しておくことが望ましい。照合の現段階において、一致する部分登録ビットパターンがなく、最大または最小の部分登録ビットパターンもないときには、最大または最小の部分登録ビットパターンが発生する段階まで遡るバックトラックが発生する。そのときに、候補アドレスが記憶されていれば、

出力工程において、候補アドレスを含む照合継続情報を出力することにより、バックトラックにより照合を継続する段階まで最短の処理時間で到達できる。

【００２４】

範囲一致検索において、最大または最小の部分登録ビットパターンに続く現段階のマスクデータのアドレスを含む照合継続情報が与えられたときは、現段階のマスクデータに示された最大または最小の部分登録ビットパターンに続く次の段階のマスクデータのアドレスを辿ることにより、被検索ビットストリングよりも値が小さくて最大の登録ビットパターン（最大の登録ビットパターン）、または、被検索ビットストリングよりも値が大きくて最小の登録ビットパターン（最小の登録ビットパターン）に到達することができる。したがって、その状態では、全照合工程および判定工程をバイパスすることができ、出力工程および出力手段では、現段階のマスクデータに対応するアドレステーブルから、現段階のマスクデータに示された最大または最小の部分登録ビットパターンに続く次の段階のマスクデータのアドレスを含む照合継続情報を出力すれば良い。

【００２５】

このように段階毎のマスクデータの最大または最小を辿りながら最大または最小の登録ビットパターンにたどり着く処理は、エントリの登録、削除がない限りマスクデータが決まればたどり着く登録ビットパターンは同一になる。したがって、マスクデータと共に、そのマスクデータに示された最大また最小の部分登録ビットパターンに対応して決まる最大または最小の登録ビットパターンを示すバイパスデータをメモリに記憶しておくことが望ましい。最大または最小の部分登録ビットパターンに続く現段階のマスクデータのアドレスを含む照合継続情報が与えられたときは、全照合工程、マスクロード工程および判定工程をバイパスし、出力工程および出力手段において、現段階のマスクデータに対応するバイパスデータを含む最終照合情報を出力して被検索ビットストリングの照合を終了することができる。したがって、中間の段階のマスクデータを辿る処理を省くことができる。

【００２６】

特に、上述したバックトラックが発生したときでも、その後の処理は候補アド

レスに示されたマスクデータに対応するバイパスデータを出力するという処理で終了し、ほとんどバックトラックによるペナルティーは発生しない。

【００２７】

上述したように、本発明の照合方法および照合装置においては、登録ビットパターンを追加あるいは削除する際に、エントリの移動は発生せず、単に、マスクデータおよびアドレステーブルを更新することにより、登録ビットパターンを追加または削除できる。したがって、照合装置に、そのような更新工程を実行する更新手段を設けておくことにより、更新するためのデータをパケットとして上流から下流に流して、エントリの追加削除を行うことができる。

【００２８】

本発明の照合方法および照合装置は、さらに、検索対象となる被検索ビットストリングのビット長増に対してもハードウェア自体のアルゴリズム的な変更は必要なく、ユニットを追加したり、一連の処理を繰り返すことで対応できるので、複数の照合段階あるいは照合装置をパイプライン方式で接続することにより、検索時間の実質的な短縮を実現できる。すなわち、複数の照合段階または照合装置の間では、照合継続情報を含むデータをパケット化して伝達することにより、データフロー型の処理で検索処理を実行できる。

【００２９】

複数種類の被検索ビットストリングを対象とする検索を実行する必要がある場合は、照合装置を複数有する分類装置において複数種類の検索を順番に実行しても良い。さらには、複数の分類装置を備えた検索装置を提供して、複数の分類装置により分類処理を並列に実行してもよい。もちろん、複数種類の被検索ビットストリングを連結して１つの被検索ビットストリングを用意し、照合を行うことも可能であるが、下流におけるマスクデータの数が膨大になる可能性があり、実際的でない。照合により得られる最大または最小の登録ビットパターンが複数の分類結果を含む範囲を示している場合は、複数の分類結果の論理積を演算し、最終の分類結果を得る必要がある。この論理演算工程および論理演算手段においては、複数の分類結果の論理積を行列演算し、最終の分類結果を得ることができる。このような論理演算工程および手段により、処理速度を向上でき、構成を簡易

にできる。

【００３０】

分類装置において、さらに処理速度を向上するのに最も効果的であると考えられるのは、メモリに対してデータを入出力するキャッシュ装置である。そして、マスクデータおよびアドレステーブルを含む照合テーブルのサイズとキャッシュ装置のキャッシュラインサイズが同一であることが望ましいことは上述した通りである。さらに、分類装置で異なる検索を行えるように、最初の照合テーブルを指定する検索種別情報を受信する手段を有する場合は、キャッシュ装置は、それぞれの検索種別および照合段階の単位でキャッシュラインを割り当て、検索種別および照合段階の単位でキャッシュラインを管理する管理手段を備えていることが望ましい。あるいは、メモリでは、それぞれの検索種別および照合段階の単位で割り当てられた個別アドレス領域に照合テーブルが記憶し、キャッシュ装置のキャッシュラインは、個別アドレス領域単位で割り当てられていることが望ましい。ビットストリングの検索においては、全体が同一または上位が同一のビットストリングが供給されることが多い。このため、従来のＣＰＵ用のキャッシュシステムのように、メモリ上のデータの局所性を期待したキャッシュシステムを採用するよりは、検索種別および段階に区分けして、その単位で過去の参照テーブルを保存するキャッシュ装置の方が、キャッシュヒット率が高く、検索あるいは照合処理速度の向上に効果がある。

【００３１】

また、キャッシュ装置としては、照合テーブルがオンキャッシュでなければ照合装置にその旨を通知し、その照合装置の処理を中断して待ち行列に戻す、要求応答型のデータキャッシュも有効である。照合テーブルがオンキャッシュでないときに、照合装置、データキャッシュおよびメモリが、そのアクセスに占有されてしまい、処理が遅れることを防止できる。パイプライン的に照合装置が接続されている場合は、１つ遅れると、その遅れは後段に影響するからである。また、被検索ビットストリングに識別情報を付加することにより、入出力の順序が保障される必要はなくなるので、待ち行列で待機させても照合処理には影響がない。。被検索ビットストリングに識別情報を付加して照合する方法は、範囲一致検索

において、バックトラックが発生したときにも有効である。

【００３２】

また、全体が同一の被検索ビットストリングが供給される頻度も高い。したがって、分類装置の分類結果を記憶した履歴キャッシュ装置であって、被検索ビットストリングを照合装置に供給する前に、当該分類装置の分類結果と被検索ビットストリングとを比較する履歴キャッシュを設けることにより、検索あるいは照合処理速度を向上できる。

【００３３】

ビットストリングの検索あるいは照合を高速で処理することができる本発明は、デジタルデータを取り扱うすべての処理で有用である。ＭＰＥＧなどの画像データを示すビットストリームから特定のビットストリングを検索したり、膨大な文字データから特定の条件を検索するなどの多様なアプリケーションが考えられるが、特に高速性が要求されるコンピュータネットワークを対象とするデータ管理装置の分野に有効である。たとえば、分類装置や検索装置の照合あるいは検索により特定されたルートにパケットを送出する機能を備えた、高速ルーティングが可能なデータ管理装置あるいは管理方法を提供できる。また、ルータやサーバなどで必要となるパケットフィルタリングあるいはＳＰＤ検索といった範囲検索を高速で実行できるデータ管理装置を提供できる。さらに、ＴＣＰパケットのリアセンブルのように、データブロックの先頭アドレスおよび末尾アドレスが一致するだけでなく、前後および包含関係になることもあるフラグメント化されたデータブロックを再構成するアプリケーションにおいても有効である。

【００３４】

さらに、本発明の照合方法および照合装置は、大規模なデータベースから有用なルールあるいはパターンを高速に発見するデータマイニングと呼ばれる分野でも有効であり、応用範囲は非常に広い。

【００３５】

【発明の実施の形態】

以下では、本発明に係るビットストリングの検索装置をルーティングテーブル検索に使用したルータ（ルーティング装置）について説明する。図１に示すルー

タ１は、ネットワーク上のパケットデータを管理する装置であり、パケットφｐの入出力を制御および管理するパケット管理機能３と、パケットφｐの転送先を判断するためのルーティング検索機能４と、パケットφｐを転送する条件を判断する条件判断機能９とを備えている。ルーティング検索機能４は、入力されたパケットの経路を示すエントリを複数備えたルーティングテーブルが記憶されたメモリ（ＳＤＲＡＭ）１１と、検索装置１０と、検索装置１０に対して検索種別を設定するなどの機能を備えた制御ユニット１２とを備えている。ルーティング検索機能４では、パケットφｐに含まれる検索対象のビットストリング（被検索ビットストリング）φｏ、たとえば、ＩＰヘッダの受信先アドレス（ＤＩＰ）をメモリ１１に記録されたルーティングテーブルの各エントリと一致検索を行う。そして、その結果φｒに基づき、パケット管理機能３が一致したエントリに設定された経路（インタフェース）へパケットデータを送出する。

【００３６】

条件判断機能９は、さらに幾つかの機能に分割されている。たとえば、ネットワーク上のセキュリティーサービスを提供するＩＰセキュリティー（ＩＰｓｅｃ）機能８と、ネットワーク上のクオリティーサービスを提供するＱｏＳ機能７と、ファイアウォール機能６などがある。これらの条件を判断する機能の多くは、パケットφｐを管理するＩＰヘッダのビットストリングを被検索ビットストリングφｏとして、その被検索ビットストリングφｏが属する範囲を検索することが要求される。ＩＰｓｅｃ（ＳＰＤ）機能８では、制御ユニット８ｃの制御のもと、検索対象として供給された被検索ビットストリングφｏと、メモリ８ｍに記憶されたＳＰＤテーブルとを検索装置８ｓにより照合する。ＱｏＳ機能７では、制御ユニット７ｃの制御のもと、被検索ビットストリングφｏと、メモリ７ｍに記録されたフロー検索テーブルとを検索装置７ｓにより照合する。さらに、ファイアウォール機能６では、制御ユニット６ｃの制御のもと、被検索ビットストリングφｏと、メモリ６ｍに記録されたパケットフィルタリングテーブルとを検索装置６ｓにより照合する。

【００３７】

パケット管理機能３は、ネットワークを介してパケットφｐを受信および／ま

たは送信可能な１つまたは複数のパケット入出力部２ａと、受信したパケットφｐをキューイングするメモリ２ｂとを備えている。キューイングされたパケットφｐは、ルーティング検索機能４でルーティング先が決まり、ＳＰＤ機能８、ＱｏＳ機能７およびファイアウォール機能６などにおける処理が決定された後に、それらの機能で決定された適切なタイミングで、パケット管理部３の何れかのパケット入出力部２ａから適切なルートあるいはネクストホップに送出される。また、ＳＰＤ機能８などの照合結果によってはキューイングされたパケットφｐが出力されずに破棄されることもある。

【００３８】

本発明の照合方法を採用した検索装置は、これらの各機能４、６、７および８の検索装置１０、６ｓ、７ｓおよび８ｓとして適用可能であり、ＩＰｖ６に基づいて被検索ビットストリングφｏのビット長が増しても高速でサーチし結果φｒをパケット管理部３に返すことができる。条件判定機能９に含まれる機能６、７および８の検索装置６ｓ、７ｓおよび８ｓにおいては、被検索ビットストリングφｏが属するルールを見出すために範囲一致検索あるいは照合が実行される。一方、ルーティング検索機能４においては、ルートを特定するために完全一致検索または最長一致検索が実行される。ルーティング検索機能４において、プレフィックス以降の最長一致の結果を得る最長一致検索は、プレフィックス以降のすべてのビットを最小側と最大側に展開した範囲一致検索に還元される。なお、本明細書において、バイナリビットツリーの「０」の方向を最小側（ＭＩＮ）と呼び、「１」の方向を最大側（ＭＡＸ）と呼ぶ。

検索装置の基本原理

以降においては、本発明に係る検索装置の概要を説明する。本発明の検索装置は複数種類の検索を単一のハードウェアで行えるものであり、いずれの機能の検索装置としてもほとんど構成を変えずに適用可能である。したがって、以降においては検索装置１０を、完全一致検索および範囲一致検索が可能な装置として代表して説明する。

【００３９】

検索装置１０は、バイナリビットツリーを構成するエントリ群と被検索キー（ビットストリング）の間の大小判定アルゴリズムを用い、エントリ（登録ビットパターン）群が表現する範囲（ルール）のどこに被検索キーが属するかを判定するハードウェアである。バイナリビットツリーに基づく照合を実行するアルゴリズムをハードウェアで実現するにあたり、バイナリビットツリーを短いビット長のテーブルに分け、多段階で照合を行う。各照合段階では、各段のテーブル内の大小判定を並列ビット照合により行う。多段階の照合にすることにより、検索対象となるビットストリングのビット長の制限はなくなり、汎用的な利用が見込まれる。また、多段階の照合にすることにより、各段階で比較するビット長を限定でき、予めすべてのエントリが登録されているものとして並列に照合しても処理時間への影響はなく、ハードウェアが増大することも無い。そして、各段に登録されているエントリを示すテーブルを示すデータ（以降ではマスクデータ）は圧縮することが可能であり、バイナリビットツリー作成（登録・削除）コストが軽減され、多段テーブルによる圧縮効果と相まって必要メモリ量も抑えられる。

【００４０】

照合過程の各段階における並列ビット照合は逐次実行プロセッサ上のソフトウェアでは実現不可能であり、並列処理ハードウェアあるいは並列実行型のプロセッサで実行することになる。並列ビット照合に必要とされるハードウェアは簡易なものであり、並列実行型のプロセッサを用いずに、経済的、コンパクトに実現することができる。

【００４１】

バイナリビットツリーによる同値および大小の判定

以下で述べる範囲検索において、右倒れ型近似ロジック、左倒れ型近似ロジックはハードウェア設計時にどちらかを選んで設計する。どちらを選んでも等価な範囲検索が可能である。両方式を切り替えられるよう設計することも可能である。一度に両方の近似で検索することも可能である。本明細書では、特に記載しない限り、右倒れ型近似ロジックに基づき設計された装置により説明する。

【００４２】

図２に、バイナリビットツリー上での完全一致検索のロジックを示してある。

ツリーの１段が１ビットに相当するバイナリビットツリーにおいては、上位から１ビットずつ被検索キーとエントリとの間で照合していく。この上位とは照合を開始するビットであり、必ずしも最上位のビット（ＭＳＢ）でなくても良く、最下位のビット（ＬＳＢ）であっても良い。バイナリビットツリーのあるノードでの次ノードへの接続は「両方向あり」「０方向のみあり」「１方向のみあり」の３通りしかない。１ノードが１エントリを意味するバイナリツリー（２分木）とは異なり、バイナリビットツリーではノードはエントリの１ビットを意味するに過ぎない。エントリは全て等ビット長なのでツリー上でのノード段数も等しく、上位ノードからの接続が存在するノードを辿っていく途中で下位ノードに対する接続が「両方向なし」となることはありえない。

【００４３】

本明細書において、図２（ａ）および（ｂ）に示すように、「被検索キー２１のビットとエントリ２２のビットが一致」となるエントリは「照合候補」と定義する。また、図２（ｃ）および（ｄ）に示すように、「被検索キー２１のビットとエントリ２２のビットとが不一致」となるエントリは「照合失格」と定義する。図２（ｅ）および（ｆ）に示すように、ノード接続が両方向である場合は、被検索キー２１のビットとエントリ２２ａのビットが一致」となるエントリ２２ａは照合候補で、「被検索キー２１のビットとエントリ２２ｂのビットが不一致」となるエントリ２２ｂは照合失格である。

【００４４】

完全一致ロジックにおいては、各ビット（各ノード）で照合をおこない、照合候補が存在する間、照合候補の次ノードを辿り、照合を継続する。照合候補が無くなればそこで照合は終了し、検索ミスヒットという最終照合結果が得られる。最終ビット（最終ノード）の照合完了後に照合候補があれば（状態が照合継続であれば）、それに対応するエントリが検索ヒットである。

【００４５】

図３は、バイナリビットツリーにおける完全一致検索の照合ロジックを表すものである。バイナリビットツリーに登録されたエントリ群２２（値０，２，８，１５）に対し、被検索キー２１ａとして値８を投入し、値８に対応するエントリが

完全一致検索ヒットと判定される例と、被検索キー２１ｂとして値４を投入し、検索ミスヒットと判定される例を示してある。最初のビットであるノードＡで被検索キー２１ａは１方向に進む。値０および２のエントリ２２は０方向、値８および１５のエントリ２２は１方向なのでエントリ２２は両方向に延び、前者のエントリ２２は被検索キー２１のビットと異なるので照合失格、後者のエントリ２２は被検索キー２１のビットと一致するので照合継続である。そして、このノードＡでの照合後に照合候補エントリが存在するため、照合継続である。

【００４６】

次のノードＥでは、被検索キー２１ａは０方向に進み、値８のエントリ２２が０方向、値１５のエントリ２２が１方向に延びている。したがって、ノードＥでも照合候補が存在するので照合継続である。ノードＦでは、被検索キー２１ａは０方向に進み、エントリ２２は片方向であるが、その値８のエントリ２２が被検索キー２１ａと同じく０方向なので照合候補が存在し、照合継続である。ノードＧでも同様であり、最終ノードである。したがって、最終ノードＧの照合後に照合候補があるので、完全一致検索は値８のエントリ２２にヒットし終了する。

【００４７】

被検索キー２１ｂでは、ノードＡでは照合候補があるため照合が継続する。ノードＢでは、被検索キー２１ｂが１方向、値０および２のエントリ２２は片方向で０方向に延びている。したがって、照合継続となっていた値０および２のエントリ２２は照合失格となり、このノードでの照合後に照合候補が残らない。このため、検索ミスヒットとなる。

【００４８】

範囲検索でも完全一致検索同様にバイナリビットツリーを上位から１ビットずつ被検索キーとエントリとの間で照合していくが、範囲検索の場合は「照合候補」に加え、「近似候補」および「近似失格」の概念がある。完全一致検索での「照合失格」が「近似候補」と「近似失格」に区別されたことになる。また、ロジック上の必要性から「近似仮定」も定義する。

【００４９】

図４に右倒れ近似の類型を示してある。図４（ａ）および（ｂ）に示すように

、「被検索キー２１のビットとエントリ２２のビットが一致」となるエントリは「照合候補」と定義する（これは完全一致検索と同様である）。この照合候補がある間、右倒れ型近似ロジックによる照合は継続する。また、図４（ｅ）および（ｆ）に示すように、「エントリ２２が両方向あり」の場合も、「被検索キー２１のビットとエントリ２２のビットが一致」である側のエントリ２２は照合候補となり、照合継続である。

【００５０】

一方、図４（ｃ）および（ｄ）に示すように、「被検索キー２１のビットとエントリ２２のビットが不一致」となるエントリは照合候補から外れる。完全一致検索ではいずれの場合も「照合失格」としていたが、右倒れ型近似の場合、図４（ｃ）のように「被検索キー２１のビットが１でエントリ２２のビットが０」となるエントリ２２を「近似候補」と定義する。逆に、図４（ｄ）のように「被検索キー２１のビットが０でエントリ２２のビットが１」となるエントリ２２を「近似失格」と定義する。近似ロジックでは、最新の近似候補を「近似仮定」として保持しておく。実際に「近似仮定」として保存が必要な値は、近似候補の指す次ノード位置である。近似候補でも近似失格でも、図４（ｃ）および（ｄ）に示すようにエントリ２２が片方向で照合候補がなければ、照合過程は終了する。

【００５１】

図４（ｅ）および（ｆ）に示すように、エントリ２２が「両方向あり」の場合は、「被検索キー２１のビットとエントリ２２のビットが不一致」となる側のエントリ２２も同様に、定義に従い近似失格あるいは近似候補となる。「両方向あり」なので「被検索キー２１のビットとエントリ２２のビットが一致」側のエントリ２２も存在し、それが照合候補となるので照合は継続する。それと共に、「近似候補」となるエントリ２２、すなわち、図４（ｅ）の「近似候補」のエントリ２２が「近似仮定」となり、それまでの「近似仮定」の値は更新される。

【００５２】

最終ビットの後に照合継続（照合候補あり）ならば、そのエントリに完全一致したことになる。途中（あるいは最終ビット）で照合終了となった場合、保持してある近似仮定（最新の近似候補）を見て近似ヒットがありえるかどうかを判定

する。近似仮定がなければ検索ミスヒットである。範囲検索の意味的には、定義された全ての範囲の外である。近似仮定があれば近似ヒットである。しかし、近似仮定が途中のノードを指す場合は、そのノードの下に複数のエントリ２２が存在する可能性があるため、近似仮定のエントリ群２２から近似決定をおこなわねばならない。この過程は、近似照合とは別のロジックで処理できる。右倒れ近似においては、以下で最大値検索（ＭＡＸサーチ）として説明している処理を行い、被検索キー２１の値以下の値のエントリの中から、最大の値のエントリ（被検索キーに近い最大のエントリ）を選択して近似エントリとして確定する。

【００５３】

図３を、バイナリビットツリーにおける右倒れ型近似ロジックによる範囲検索を表すものとすると以下のようになる。値４の被検索キー２１ｂについては、ノードＡで０方向の値０または２のエントリ２２は照合候補、１方向の値８または１５のエントリ２２は近似失格となる。ノードＡでは、照合候補が存在するため照合継続となる。次にノードＢでは、０方向の値０または２のエントリが近似候補となり、近似候補なので近似仮定（最新の近似候補を保持）の更新を行い、近似候補の指す次ノードであるノードＣを近似仮定として保持する。

【００５４】

ノードＢの照合後に照合候補がなく、照合過程はいったん終了する。近似仮定があるので、近似ヒットである。従って、近似仮定となるエントリ群からＭＡＸサーチにより近似確定をおこなう。この場合、値０と値２のエントリ２２から値２のエントリを選択することになる。図３に示したように、被検索キー２１ｂは２≦キー値≦７の範囲の値であれば、同じ結果である値２のエントリ２２を得ることができる。つまり、値２のエントリ２２は（２～７）という範囲を意味するエントリとなっている。このため右倒れ近似が範囲検索になるのである。

【００５５】

次に、値８の被検索キー２１ａにおいては、ノードＡでは、０方向のエントリ２２（値０と２が属する）が近似候補となり、近似仮定（ノードＢを指す）が更新される。１方向がエントリ２２（値８と１５が属する）が照合候補となり、照合候補も存在するため照合継続となる。ノードＥでは、ビット値０方向のエント

リ２２が照合継続、ビット値１方向のエントリが近似失格となる。近似候補はないため、近似仮定は更新されない。ノードＦ以降は単方向で照合継続となり、近似候補が発生しないため、近似仮定は更新されない。そして、最終ビット後に照合継続（照合候補あり）なので、値８のエントリ２２に完全一致するという最終照合結果が得られる。

【００５６】

さらに、値１４の被検索キー２１ｃを投入した場合を考える。ノードＡでは、ビット値０方向のエントリ２２（値０と２が属する）が近似候補となるので、近似仮定（ノードＢを指す）が更新される。また、ビット値１方向のエントリ（値８と１５）が照合候補となり照合継続となる。ノードＥでは、ビット値０方向の値８のエントリ２２が近似候補となり、近似仮定は更新される（ノードＦを指す）。また、１方向の値１５のエントリ２２が照合候補となるので、照合継続となる。ノードＨでは、単方向で値１５のエントリ２２が照合候補となり、近似候補が発生しないので、近似仮定は更新しないで、照合継続となる。ノードＩでは、単方向で照合失格となる。従って、いったん照合過程は終了する。

【００５７】

このとき、近似仮定はノードＥでの更新でノードＦを指している。近似確定のため、ノードＦ以下をＭＡＸサーチし、値８に対応するエントリ２２が近似ヒットと決定される。したがって、この被検索キー２１は、値８のエントリ２２がカバーする範囲（８～１４）に属することが判明し、範囲一致検索が実行されたことになる。

【００５８】

図５に、左倒れ近似における「照合候補」「近似候補」「近似失格」の定義を示してある。右倒れ近似と「近似候補」「近似失格」のビットが入れ替わっただけである。照合ロジックも、上記の入れ替わり以外は右倒れ近似と同様である。ただし、右倒れ型近似では近似確定するため近似仮定に対するＭＡＸサーチをおこなうのに対し、左倒れ近似の場合、近似確定のために近似仮定に対するＭＩＮサーチをおこなう点が異なる。ＭＩＮサーチでは、被検索キー２１の値以上の値のエントリの中から、最小の値のエントリ（被検索キーに近い最小のエントリ）

を選択して近似エントリとして確定する。

【００５９】

最終ビットまで「エントリのビットと被検索キーのビットが一致」であった場合、つまり最終ビットの照合後に照合継続であった場合、これも範囲検索としてはヒットとする。つまり、右倒れ型近似による範囲検索とは、完全一致ヒットと近似一致ヒットを区別せず、「キー≧エントリ」となる最大エントリを検索するものである。ただし、検索の用途によっては「＝」を含まない、「キー＞エントリ」の右倒れ近似とすることもできる。また、「＝」一致なのか「＞」一致なのかを区別した結果とすることもできる。

範囲検索におけるエントリデータの形式

右倒れ型近似による範囲検索では、領域は右倒れ近似ロジックに適合するよう正規化されエントリ登録される。この正規化は領域を表す２値（［ＦＲＯＭ，ＴＯ］）のうちＴＯ（大値）側の補正である。例えば［ａ，ｂ］で表される領域Ｒは値ａのエントリＡと値ｂ＋１のエントリＢとして登録される。エントリＡは領域Ｒを表現し、エントリＢは領域Ｓ［ｂ＋１，∞］を表現する。エントリＡに右倒れ型近似ヒットすることは、領域Ｒに範囲検索ヒットすることである。エントリＢに近似ヒットすることは、領域Ｓにヒット、つまり範囲検索ミスヒット（領域Ｒ外）を意味する。近似ミスヒットは領域Ｑ［０，ａ―１］、つまり範囲検索ミスヒット（領域Ｒ外）を意味する。

【００６０】

一点を表す領域Ｕ［ｃ，ｃ］も定義可能であり、値ｃのエントリＣと値ｃ＋１のエントリＤとして登録される。エントリＣは領域Ｕを表現し、エントリＤは領域Ｖ［ｃ＋１，∞］を表現する。値ｃと値ｃ＋１の隣接するエントリが登録されるため、値ｃであるエントリＣへのヒットは完全一致ヒットでしかありえず、点領域Ｕに範囲検索ヒットを意味する。エントリＤへの近似ヒットは領域Ｖにヒット、つまり範囲検索ミスヒット（領域Ｕ外）である。近似ミスヒットは領域Ｔ［０，ｃ―１］、つまり範囲検索ミスヒット（領域Ｕ外）を意味する。

【００６１】

左倒れ型近似でも領域の正規化はおこなわれる。ただし、右倒れ型とは逆にＦＲＯＭ（小値）側を補正する。

【００６２】

エントリと集合の関係を、エントリＩは集合ＳＥＴ（Ｉ）［ｉ，∞］を意味すると定義する。このとき、領域Ｒ［ａ，ｂ］は、値ａのエントリＡすなわち集合ＳＥＴ（Ａ）［ａ，∞］と、値ｂ＋１のエントリＢすなわち集合ＳＥＴ（Ｂ）［ｂ＋１，∞］との差集合ＳＥＴ（Ａ）－ＳＥＴ（Ｂ）で表せる。エントリＡに右倒れ近似ヒットするということは、エントリＡにヒットし、エントリＢにミスヒットということである。

【００６３】

照合方法

図６に示すように、多ビットのバイナリビットツリー２４を多段階または多階層に分けて、さらに各段階を適当な数のエントリが表れるビット幅に分割することにより、テーブル化できる。一定単位の短ビット長の分割テーブルまたは部分テーブル２５を規定することにより、多ビットのバイナリビットツリーを、上位の部分テーブル２５から指定される各段階の部分テーブル２５の組み合わせにより表現できる。そして、固定ビット数の部分テーブルにすることで、並列処理対象エントリは有限固定となり、部分テーブル２５に含まれるエントリをハードウェアにより並列照合することができる。さらには、部分テーブル２５に含まれる最大エントリ数は有限固定になるので、実際に部分テーブル２５に含まれるエントリの有無にかかわらず、被検索キーが部分テーブル２５に含まれ得るエントリのいずれと一致するかを、ある部分テーブル２５が選択されると自動的にハードウェアで照合することも可能となる。

【００６４】

図６では、３ビットの部分テーブル２５によりバイナリビットツリー２４を表現しており、各部分テーブル２５に収容される部分エントリ２６は値０～７の８エントリが最大数である。ハードウェア実装としては、メモリのワードのビット長や検索キーのビット長などを考慮し適切なテーブルビット長を決定するが、テーブルビット長がｎビットになると並列処理対象であるテーブル内のエントリ数

が2^nになるため、一般的には４ビットテーブル程度が適切である。

【００６５】

部分テーブル化することにより、各段の部分テーブルでの検索は有限固定エントリ数でのサーチになる。また、被検索キーが固定ビット幅であれば、エントリを示すバイナリビットツリー２４は固定長となり、それを表現する部分テーブル２５の段数は予め定まった有限固定段数となる。このため、一致検索においてはバックトラックの発生しないバイナリビットツリー検索となるので、検索に要するワースト時間はエントリ数にかかわらず一定時間となる。

【００６６】

図７（ａ）に示すような論理イメージで、バイナリビットツリー２４は部分テーブル化されている。図７（ｂ）は、部分テーブル２５を汎用テーブル２７として構成した様子を示してある。この汎用テーブル２７では、１エントリはビットパターン値２７ａと、それに対応する下位テーブル２７へのポインタ２７ｂで構成され、それが最大ｎ個あるテーブル構造となる。さらに、図７（ｃ）に、汎用テーブル２７を圧縮したテーブル２８で部分テーブル２５を実現した様子を示してある。この圧縮テーブル２８は、エントリの有効無効を示すビットフラグ（マスクデータ）２８ａと下位へのポインタ２８ｂで１エントリが構成される。下位の圧縮テーブル２８へ続くポインタ２８ｂは、圧縮テーブル２８が記憶されているＲＡＭのアドレスなどになり、アドレステーブルを生成する。

【００６７】

汎用テーブル２７と圧縮テーブル２８の違いは、汎用テーブル２７では登録ごとにエントリにビットパターンをセットしていくのに対し、圧縮テーブル２８では、テーブルは固定ビット幅のテーブルであり、ビットパターンを省略している点である。部分テーブル２５が固定ビット幅の場合、ビット幅に対応して上限エントリ数が決まる。したがって、例えば４ビットテーブルの場合、エントリＮｏ０はビットパターン００００、エントリＮｏ１はビットパターン０００１というようにエントリＮｏとビットパターンの関係を一対一に固定できる。したがって、アクティブなエントリＮｏを示すことだけで、部分テーブル２５に含まれる部分エントリ２６を指定することができる。エントリＮｏを示すデータはいくつか

考えられるが、図７（ｃ）に示すようにエントリＮｏに対応したビット位置にオンオフのフラグとなるビットを配置したビットパターン（マスクパターン）２８ａはハードウェアで利用しやすい形態の情報である。また、マスクパターン２８ａにより部分エントリの有無を示すことにより、実際の部分エントリのビットパターンを登録する処理および登録するためのハードウェアおよびソフトウェアを省略できる。

【００６８】

この部分テーブル２５を圧縮テーブル２８とした最適化により、部分テーブル２５をそれに対応する被検索キーの部分（部分被検索キー）と照合するハードウェアが単純化される。４ビットの部分テーブル２５であれば、４ビットの部分エントリ、すなわち、値０から１５までの部分エントリが取りうる値と部分被検索キーとの照合を並列に行うハードウェアを採用することにより、照合作業およびハードウェアが単純化される。すなわち、全エントリハードウェア並列照合が可能となり、それが単純化される。また、エントリＮｏとビットパターンの関係が固定でソート済なので、エントリ（部分エントリ）の追加削除は、基本的にはマスクデータ２８ａを更新するだけで実現できる。そして、テーブル内でのＭＩＮ／ＭＡＸサーチはビットパターンの比較なしにエントリＮｏのみでおこなえる。したがって、範囲一致検索において、近似候補となる部分テーブル２５までバックトラックが発生したとしても、その後は、エントリが有効か無効かのフラグ（マスクデータ）２８ａの参照だけでテーブル内におけるＭＩＮ／ＭＡＸサーチは終わることができるので、実質的にはバックトラックが発生しない照合方法ということも可能である。

【００６９】

ＭＩＮ／ＭＡＸサーチとそのバイパス

ＭＩＮ／ＭＡＸサーチの原理

バイナリビットツリーにおけるＭＩＮ／ＭＡＸサーチの原理は以下の通りである。図８（ａ）に、右倒れ近似におけるＭＡＸサーチの概要を示してある。ビット位置（ノード）Ａ１にて、被検索キー２１に対してエントリ２２がなくなり、右倒れ型範囲検索の照合候補がなくなる。そのため、照合は途中で終了し、近似

仮定エントリに対するＭＡＸサーチを開始する。エントリ２２が片方向しかない場合はその方向に辿る。ノードＡ２においては、０方向にしかエントリ２２がないので、それを辿る。次のノードＡ３においては、エントリ２２が０および１の両方向にあるので、大きい方向（０＜１なので１の方向）を優先して辿る。各ノードに対して同様の処理を行い、最下段ビット位置Ａ６まで辿り着き、得られるエントリ２２がＭＡＸエントリ（被検索キーの値よりも小さくて最大の値をとるエントリであり、被検索キーに近い最大のエントリ）である。

【００７０】

図８（ｂ）は、左倒れ近似におけるＭＩＮサーチの概要を示してある。左倒れ型範囲検索の場合、同種のロジックで小さな方向（０の方向）を優先して辿り、ＭＩＮエントリ（被検索キーの値よりも大きくて最小の値をとるエントリであり、被検索キーに近い最小のエントリ）を取得するＭＩＮサーチをおこなう。

【００７１】

テーブルにおけるＭＩＮ／ＭＡＸサーチ

バイナリビットツリーをテーブル化すると、ＭＩＮ／ＭＡＸサーチのロジックもテーブル対応にしなければならない。図９（ａ）に汎用テーブルでのＭＩＮ／ＭＡＸサーチのうち、右倒れ型近似におけるＭＡＸサーチを示してある。テーブルＡ１にて照合失格（キー値２１と同値のエントリ２２が無い状態）となる。テーブル内で「キー値＞エントリ値」となるエントリの中から、最大値であるエントリ（「キー値＞エントリ値」の中でエントリ値が最もキー値に近いエントリ）を選択し、近似候補エントリ２２ｘとする。近似候補２２ｘの指す次テーブルが近似仮定エントリ２２ｙである。そして、その近似仮定エントリ２２ｙ以下に対するＭＡＸサーチを開始する。

【００７２】

まず、近似仮定エントリ２２ｙのテーブルであるテーブルＡ２に移る。テーブルＡ２内のエントリを調べ、最大値をもつエントリを選択する。そのエントリの次段テーブルに移り、以下0、最大値を持つエントリ選択と下段への移動を繰り返し、最下段まで到達する。最下段でも最大値エントリを選択し、最大値エントリの登録データに辿りつき、ＭＡＸエントリを出力する。図９（ｂ）は、左倒れ

型範囲検索の場合であり、同種のロジックでＭＩＮサーチをおこなう。

【００７３】

図１０に、圧縮テーブル２８のＭＩＮ／ＭＡＸサーチのロジックのうち、右倒れ型近似に対応したＭＡＸサーチのプロセスを示してある。まず、圧縮テーブル２８ではマスクデータ２８ａにおいてエントリＮｏとビットパターンの関係が固定であり、ソート済の状態に等しいので、テーブル内での最小値／最大値エントリの選択はビットパターンの比較なしに、マスクデータ２８ａの対応するエントリＮｏにフラグを立てるだけで行うことができる。つまり、有効無効フラグの参照だけでテーブル内における最小値／最大値エントリの選択は終わる。そして、追加したエントリＮｏに対応するアドレステーブル２８ｂのレコードにそのエントリＮｏに続く圧縮テーブル２８のアドレスを記録することにより圧縮テーブル２８をリストで繋ぐことが可能となり、リニアアドレス上のテーブルでなくてもバイナリビットツリーの多段階検索が可能となる。

【００７４】

右倒れ型範囲検索におけるＭＡＸサーチを説明する。テーブルＡ１のマスクデータ２８ａにおいて照合候補がなくなり、照合が終了すると、近似仮定エントリに対するＭＡＸサーチを開始する。近似候補の選択は、テーブルＡ１のマスクデータ２８ａの内でキー値＞エントリ値となるエントリのうち最大値をもつエントリ（最大のエントリ）を選択することでおこなう。ビットマップによる圧縮テーブルなので、エントリＮｏ＝エントリ値であり、上記条件に合致するエントリはキー値エントリから左にみていって最初に１が立っているエントリが近似候補２２ｘである。この例では照合が終了したのと同じテーブルＡ１に近似候補エントリ２２ｘが存在する。近似候補エントリ２２ｘの指す次テーブルが近似仮定エントリ２２ｙである。

【００７５】

したがって、近似仮定エントリ２２ｙであるテーブルＡ２に移動する。テーブルＡ２内で最大値をもつエントリを選択する。エントリＮｏ＝エントリ値なので、右端からみて最初に１の立つエントリが最大値エントリ２２ｍである。最大値エントリ２２ｍの指す次段テーブルＡ３に移り、エントリ選択と下段への移動を

繰り返し、最下段エントリまで到達する。最下段でも同様に最大値エントリ２２ｍを選択し、最下段での最大値エントリ２２ｍの指すデータに到達する。これにより最大のエントリを得ることができる。左倒れ型範囲検索では、同種のロジックでＭＩＮサーチを行う。

【００７６】

ＭＩＮ／ＭＡＸサーチのバイパス

圧縮テーブル２８ではエントリＮｏとビットパターンの関係が固定でソート済なので、ＭＩＮ／ＭＡＸサーチはビットパターンの比較なしにエントリＮｏのみでおこなえる。有効無効フラグの参照だけでテーブル内におけるＭＩＮ／ＭＡＸサーチは終わる。しかしながら、エントリの登録・削除がない限り、下位の部分テーブル２８を辿って到達する最終的な最大のエントリあるいは最小のエントリは不変である。したがって、毎回、ＭＩＮ／ＭＡＸサーチをするのは無駄であり、図１１（ａ）および（ｂ）に示すように、各テーブルにバイパスＴＡＧ（ＢＰ）を設け、そのテーブル２８以下でＭＩＮ／ＭＡＸサーチした結果、すなわち、最大なエントリあるいは最小なエントリを記録しておくことによりＭＩＮ／ＭＡＸサーチを省略することが可能となる。これにより、最大のエントリ、あるいは最小のエントリを検索するプロセスは、近似仮定した圧縮テーブル２８にアクセスするだけのプロセスに代わり、検索時間は大幅に短縮される。

【００７７】

したがって、近似仮定したエントリを近似確定させる際、近似仮定エントリ２２ｙの指すアドレステーブル２８ｂのアドレス（以降においてはＴＡＧ（次段テーブルを指すＴＡＧ）と称することがある）を使い、次段テーブルのバイパスＴＡＧ（ＢＰ）により近似確定する最大または最小のエントリを決定することができる。範囲検索の論理として右倒れ型でおこなう場合はバイパスＢＰには最大のエントリが記録され、左倒れ型でおこなう場合はバイパスＢＰには最少のエントリが記録される。複数の範囲検索系があった場合、照合のロジック構成としては右倒れ型か左倒れ型かのいずれかに統一する。バイパスＢＰに両値を記録することにより、テーブルに記録する情報量の増加、照合用のロジックの追加、エントリの登録削除用のロジックの追加などが発生するが、左右両方式を混在させるこ

とも可能である。

【００７８】

さらに、バイパスＢＰを記録することにより、ＭＩＮ／ＭＡＸサーチにおけるバックトラックの遅延も解消することができる。ＭＩＮ／ＭＡＸサーチとは、近似候補エントリ群から近似結果を確定させるための検索である。ところで、右倒れ型であっても左倒れ型であっても、テーブル型ビットツリーの場合、照合継続中に毎段で近似候補が更新されていくわけではない。ある段で近似候補が更新され、その後、数段の近似候補更新を伴わない照合継続があり、下段のほうで照合失敗となった段階で、上段で更新された近似候補を使い、結果を確定するためにさらにＭＩＮ／ＭＡＸサーチをおこなう必要がある場合がある。

【００７９】

図１２に一例を示してある。上位のテーブル２８から下位のテーブル２８に照合継続し、照合失格となった段階で、近似仮定に基づき上位のテーブル２８に戻り、別ロジック（ＭＩＮ／ＭＡＸサーチ）で再びツリー状にテーブル２８を降りていくことになる。これはバックトラックということができる。照合失敗によりＭＩＮ／ＭＡＸサーチでツリーを辿るロジックが変わるので、ＭＩＮ／ＭＡＸサーチでは再度のバックトラックは発生しないが、照合失敗位置と近似仮定位置が離れている場合、バックトラックと目されるＭＩＮ／ＭＡＸサーチのコストは重い。

【００８０】

これに対し、図１３に示すように、バイパスＢＰを記録しておけば、一度バイパスＢＰの値を読むことにより、最大または最小のエントリが確定するため、実質的にはサーチプロセスとなるような時間の係る処理は不要となり、実質的にはバックトラックの発生を抑えられる。図１３は、右倒れ型の例であり、テーブルA1で近似仮定更新・照合継続となり、テーブルＡ２では近似仮定更新無し・照合継続となり、テーブルＡ３で近似仮定更新無し・照合失敗となる。このとき、テーブルＡ３での照合失敗で近似確定のため、ＭＡＸサーチに移行するが、最新の近似仮定はテーブルＡ２´であるため、テーブルＡ２´にアクセスする。そして、テーブルＡ２´にバイパスＢＰが記録されていれば、それにより直接、最下

段の最大のエントリまで移動でき、照合過程は終了する。

【００８１】

検索装置、分類装置およびバイナリビットツリー照合器

図１４に検索装置１０の概要を示してある。この検索装置１０は、各種の検索を行う分類器１５を複数備えており、各種の検索を並列に行えるようになっている。もちろん、１つの分類器１５により各種の検索をシーケンシャルに行っても良い。検索装置１０に対しては上流のアプリケーション、たとえば、図１に示した制御ユニット１２、８ｃなどから被検索ビットストリングであるキー２１と、その検索種別を示すルートＴＡＧ３１と、検索ジョブを識別する情報（ＩＤ）３２とが供給される。たとえば、ＳＰＤ検索機能８であれば、１つのパケットデータφｐを管理するために、ソースＩＰ、デスティネーションＩＰ、ソースポート、デスティネーションポートおよびプロトコルを示すビットストリングを少なくとも照合して、それらの全てを満足するルールを検索する必要がある。このような場合は、ソースＩＰ、デスティネーションＩＰ、ソースポート、デスティネーションポートおよびプロトコルを示すビットストリングをそれぞれ異なる分類装置１５に対して被検索ビットストリング（被検索キー）２１として供給し、それぞれのビットストリングと比較するビットパターン（エントリ）２２が登録されたバイナリビットツリーの最上位の部分テーブル２５（２８）を示すメモリ１１あるいは８ｍのアドレスをルートタグ３１として供給する。これにより各々の分類装置１５においては、供給された被検索キー２１を、その被検索キー２１の範囲検索に用いるべく予め登録されているビットパターン（エントリ）２２と照合することができる。

【００８２】

さらに、検索装置１０は、複数の分類装置１５に供給された被検索キー２１の照合結果により得られるルールの論理積を求めて最終結果を取得するための論理演算装置（ＡＮＤ化器）１９を備えている。各々の分類装置１５における検索処理時間は、各々の分類装置１５に供給される被検索キー２１のビット長により異なり、さらに、照合過程において上位の部分テーブル２８に戻るバックトラックが発生するか否かによっても異なる。したがって、分類装置１５から同期して検

索結果が出力されない可能性がある。このため、分類装置１５に供給される被検索キー２１にＩＤ３２を付加することにより、論理演算装置１９における論理演算を可能とし、さらに、パケット管理装置３において管理されるパケットデータφｐとの関連付けも可能なようにしている。

【００８３】

本例の分類装置１５は、パイプライン状に接続された複数の照合装置５０を備えており、これらの照合装置５０の間ではデータパケットとして標準化された照合継続情報（入出力パラメータ）３３が伝達され、パケット駆動される構成となっている。したがって、分類装置１５は複数の照合装置５０によりデータフローが形成されており、クロック同期のハードウェアで実現することも可能であるし、データフロー型のハードウェアで実現することも可能である。分類装置１５は、さらに、圧縮された部分テーブル２８が記録されたメモリ１１と照合装置５０との間の入出力速度を向上するためのバースト転送バッファ１６も備えている。

【００８４】

以下で説明する検索装置１０、分類装置１５および照合装置５０の構成は、ソフトウェアで実現することも可能であるし、ソフトウェアを主体として部分的にハードウェアを用いて実現することも可能である。しかしながら、実行速度を考慮するとできるかぎりハードウェアで実現することが望ましい。本発明に係る部分テーブル２５あるいは２８を用いた検索方法は、単一、あるいはほぼ単一な構成の照合装置（照合器）５０をパイプライン状に接続することによりビット長の変化に対応でき、また、ルートタグ３１といった検索種別を示す情報が提供されることにより単一な構成の照合装置５０により多種類の検索を実行できるようになっている。したがって、本発明の照合方法は、非常にハードウェア化しやすい照合方法であり、経済的で、処理速度の速い検索装置を提供できる。また、上位のソフトウェアあるいはアプリケーションからは、被検索キー２１と、検索種別情報３１を提供するだけで、所望の検索を実行できるので、上位のアプリケーションの負荷も非常に小さい。

【００８５】

図１５に、８個の４ビット型の照合器５０をパイプライン状に繋いで３２ビッ

トの被検索キー２１に対する検索を行う分類装置１５の概略構成を示してある。照合器ハードウェアの構成である。照合継続情報３３に含まれるルートタグ（RootTAG）３１および照合タグ（NextTAG）３４は、ＳＤＲＡＭメモリ１１の上の圧縮テーブル２８を指すポインタ（アドレス）である。各照合装置５０は、照合継続情報３３に含まれる被検索キー２１から順番に４ビットずつを部分被検索キー３６として受け取り、ルートタグ３１または照合タグ３４により指定されるテーブル２８からマスクデータ２８ａを読み取り、近似一致による比較をおこなう。比較によりヒットしたエントリ（部分エントリ）２６に対応する次段の圧縮テーブル２８のポインタ(NextTAG)３４を、圧縮テーブル２８のアドレステーブル２８ｂから読み取り、照合継続情報３３に含めて次の照合装置５０に供給する。

【００８６】

被検索キー２１や照合タグ３４などの照合継続情報３３は、別々に供給されても良い。しかしながら、データパケットとしてまとめて入力することにより、データフロー化しやすい。また、検索対象となる部分被検索キー３６は、上位のプログラムにより予め分割された状態で照合装置５０に供給されても良い。しかしながら、被検索キー２１の値は全ビットが照合継続情報３３に含められ、シフトしながら必要なビット数ずつを部分被検索キー３６として使用していくことが望ましい。これは、照合器パイプライン構成の途中でパイプラインステージを逆流（バイパス等によるバックトラック）する可能性があり、被検索キー２１を照合器５０の外部で切り分けたのでは、パイプラインの進行（クロック）同期で適切なステージ（段階）へ投入していくことができないためである。また、ＳＤＲＡＭ１１に対するアクセスの状態（競合・排他）や、データキャッシュをつけた場合のオンキャッシュかどうか等で各照合装置５０における処理速度は変動する可能性が高い。したがって、照合器パイプラインの進行自体をクロック同期ではなく要求応答型（データフロー型）とした方が処理効率は高く、この点でも、被検索キー２１を照合継続情報３３に含め、さらに、照合検索情報３３をデータパケット化して次の段階の照合装置５０に供給するメリットがある。

【００８７】

図１６には、分類装置１５の異なる例を示してある。この分類装置１５はＭＡ

Ｘ／ＭＩＮサーチを実行する際のバイパス用のバス（ＢＰバス）３９を備えており、照合装置５０は、ＢＰバス３９に出力するインターフェイスを備えている。したがって、この分類装置１５においては、各照合装置５０が非同期でテーブル２８にアクセスする頻度が上昇する。図１５に示したように、各段のテーブル２８を同一のＳＤＲＡＭ１１におくと、パイプラインを構成する各照合器５０のメモリアクセス競合があるため、パイプラインが有効に機能しない。従って、各段のテーブル２８を個別のメモリ１１ａにおき、各照合器５０のメモリアクセスを独立としたほうがパイプラインの有効性が高い。この照合装置パイプラインを備えた分類装置１５では、照合器５０の（メモリアクセスを含む）所要時間をＴとし３２ｂｉｔの被検索キー２１が途切れなく入力されれば、検索のスループットＴ（本当は１／Ｔ）、レイテンシは８Ｔである。

【００８８】

照合器５０の構成としてはＳＤＲＡＭ１１を除いて処理時間が一定していてパイプラインを組みやすい。データキャッシュ制御によりＳＤＲＡＭアクセスを排除する履歴キャッシュ方式や専用ＳＲＡＭ型履歴キャッシュ方式を使えばアクセス時間を一定化でき、パイプラインを構成しやすい。これに対し、共有型データキャッシュであれば、キャッシュのスループットと全パイプラインステージのスループットの合計が等しい、あるいは共有型データキャッシュの方がスループットを高くする必要がある。

【００８９】

図１７に、分類装置１５のさらに異なる例をしめしてある。上記の例では、複数の照合器５０を処理パイプラインとして並べた構成であるが、省リソースを優先した場合は、図１７に示したように、１個の照合器５０を繰り返し使う構成も可能である。この場合、照合器５０からのメモリアクセスはある時点では常に１アクセスとなるので、テーブル２８を全て同一のＳＤＲＡＭ１１に置くことで処理速度の優劣はない。

【００９０】

照合装置の構成

図１８に照合装置５０の構成を示してある。この照合装置５０は、被検索ビッ

トストリング（被検索キー）２１を複数ビットの部分被検索ビットストリング（部分被検索キー）３６に分けて、予め登録された複数の登録ビットパターン（エントリ）２２と多段階で照合するための１つの照合段階を実行する照合装置である。照合装置５０は、照合継続情報３３に含まれるコマンド３７をデコードして照合装置５０における処理を制御するコマンドデコードユニット５１と、被検索キー２１から、現段階の部分被検索キー３６を選択し、その部分被検索キー３６が取り得る全ての値と照合する比較ユニット（全照合手段）５２と、メモリ１１からその照合段階の圧縮テーブル２８のマスクデータ２８ａをロードするロードユニット５３とを備えている。ロードユニット５３は、圧縮テーブル２８をメモリ１１からバッファ５９に格納するテーブルロード部５３ａと、バッファ５９に格納された圧縮テーブル２８からマスクデータ２８ａを読み出すマスク読み出し部５３ｂとを備えている。圧縮テーブル２８にマスクデータ２８ａと共に格納されているアドレステーブル２８ｂおよびバイパスデータＢＰは多くの処理で必要になる。このため、本例では圧縮テーブル２８をバッファ５９にロードしている。マスクデータ２８ａは、上述したように、エントリ２２の部分ビットパターン２６であって、先行する段階から得られる照合継続情報３３により決まる、部分被検索キー３６に対応する段階の部分登録ビットパターン（部分エントリ）２６をフラグ（ビットパターン）により示すものである。

【００９１】

照合装置５０は、さらに、比較ユニット５１における全照合結果とマスクデータ２８ａに、部分被検索キー３６に一致する部分エントリ２６の有無、すなわち一致を示すことができる判定ユニット５４と、判定ユニット５４の照合結果により、アドレステーブル２８ｂから照合により特定された部分エントリ２６に続く次の段階のマスクデータ２８ａのアドレス（タグ（本例では圧縮テーブル２８のアドレス））を照合タグ（照合候補）３４として出力する出力ユニット５５とを備えている。この照合装置５０は、一致検索のみではなく範囲検索も行えるように、判定ユニット５４は、部分被検索キー３６に値が最も近い最大（右倒れ型近似について記載し、左倒れ型近似については、最大を最小に変更することによりほぼ対応できる）の部分エントリ２６の有無を示すことが可能である。本例の判

定ユニット５４は、比較ユニット５２の出力をマスクデータ２８ａによりマスク処理するマスク器（ＶＬＤ）５４ａと、マスク器５４ａの出力から状態を選択する選択器５４ｂとを備えている。

【００９２】

出力ユニット５５は、最大の部分エントリ２６に続く次の段階のマスクデータ２８ａのアドレス（圧縮テーブルのアドレス）を照合タグ３４として出力する。さらに、出力ユニット５５は、判定ユニット５３において、最大の部分エントリ２６が見つかったときは、その最大の部分エントリ２６に続く次の段階のマスクデータ２８ａのアドレス（圧縮テーブル２８のアドレス）を候補アドレス（候補タグ、近似候補）３５として出力し、照合継続情報３３のパケットデータに保存する。したがって、一致する部分エントリ２６があり照合継続の場合は、最大の部分エントリ２６が見つかると候補タグ３５にそれが保存され、一致する部分エントリ２６がない照合終了で、さらに、その照合段階では最大の部分エントリ２６がないときに、候補タグ３５を照合タグ３４として照合継続情報３３に出力することによりＭＡＸサーチに移行することができる。

【００９３】

さらに、照合装置５０の出力ユニット５４は、ＭＡＸサーチに移行したときに、圧縮テーブル２８からバイパスデータＢＰを読み取って最終照合結果として出力することができる機能も備えている。したがって、出力ユニット５５は、照合状態などを反映したコマンド３７を生成するコマンド更新部５５ａと、照合タグ３４を更新する照合タグ更新部５５ｂと、候補タグ３５を更新する候補タグ更新部５５ｃと、バイパス読み出し出力部５５ｄとを備えている。そして、出力インターフェイス５６からは、出力ユニット５５により更新されるパラメータ３４、３５および３７と、入力された照合継続情報３３に含まれるＩＤ３２と、被検索キー２１とを含むデータパケットが照合継続情報３３として下流の照合装置５０に供給される。

【００９４】

圧縮テーブル２８にバイパスデータＢＰが設定されていないときは、照合終了を示すコマンド３７を受け取ると、出力ユニット５５の照合タグ更新部５５ｂが

マスクデータ２８の最大の部分エントリ２６に続く次の段階のマスクデータのアドレスを照合タグ３４として出力し、ＭＡＸサーチが継続される照合装置５０のパイプラインを制御する。

【００９５】

さらに、本例の照合装置５０は、マスクデータ２８ａおよびアドレステーブル２８ｂを更新することにより、エントリを追加または削除する更新ユニット６０と、更新されたテーブル２８をメモリ１１に書き込むテーブルライトユニット６１とを備えている。更新ユニット６０の動作については、以下でさらに詳しく説明する。

【００９６】

図１９に、照合装置５０における処理をフローチャートにより示してある。まず、ステップ１０１でコマンド３７がエントリの追加削除を示しているときは、ステップ１０２で更新ユニット６０によりテーブル２８を更新する。ステップ１０３でコマンド３７から前段階の照合が近似で照合終了していることが判明すると、以下に説明する照合過程のステップ１０４、１０５および１０６をバイパスし、出力段階１１０のステップ１１１でバイパス読み出し出力部５５ｄによりバイパスデータＢＰを図１６に示したバイパスバス３９に出力して照合を終了する。

【００９７】

コマンド３７より照合を継続する場合は、ステップ１０４において比較ユニット５２により部分被検索キー３６を比較し、ステップ１０５においてロードユニット５３によりマスクデータ２８ａをロードし、ステップ１０６において判定ユニット５４により照合結果を得る。そして、ステップ１０７において一致する部分エントリ２６があれば照合継続なので、出力段階１１０のステップ１０９において候補タグ更新部５５ｃにより候補タグ３５を更新し、ステップ１１４で照合タグ更新部５５ｂにより照合タグ３４を更新し、コマンド３７をコマンド更新部５５ａにより照合継続にセットして出力する。

【００９８】

一方、一致する部分エントリ２６がなく、照合終了であるが、ステップ１０８

で近似の部分エントリ（最大の部分エントリ）２６がある場合は、出力段階１１０のステップ１１３において照合タグ更新部５５ｂが近似の部分エントリ２６に続くタグアドレスを照合タグ３４にセットして出力する。ステップ１０８で近似の部分エントリ２６がない場合は、出力段階１１０のステップ１１２において、照合タグ更新部５５ｂが、上流の照合過程で得られ、照合継続情報３３に含まれて伝達された候補タグ３５を照合タグ３４にセットしてバイパスバス３９に出力する。これにより、ＭＡＸサーチが開始される。

【００９９】

これらの工程は、実際には、独立して処理できる工程は並列に実行される。たとえば、コマンドを解釈する工程１０１および１０２、部分被検索キー３６を比較する工程１０４と、マスクデータをロードする工程１０５とは独立した処理であるので、並列に実行できる。テーブルをロードするフェーズはＳＤＲＡＭアクセスであるため時間がかかり、比較フェーズなどとは並列に実行することによりメモリアクセスのオーバヘッドを緩和できる。したがって、読み出し速度が非常に速い特殊なメモリを用いなくても本例の照合装置５０においては十分な検索スピードを確保できる。また、出力段階１１０において異なるパラメータを更新するフェーズも相互独立であり、コマンド更新部５５ａ、照合タグ更新部５５ｂおよび候補タグ更新部５５ｃも独立しており、これらにおける処理は並列に実行可能である。したがって、図２０に示すように、通常の照合モードの場合、照合装置５０においては、テーブルロード５３と比較５２などが並列で実行される。

【０１００】

本例のロードユニット５３は、照合タグ３４の指すＳＤＲＡＭメモリ１１上の圧縮テーブル２８を照合器５０の内部のテーブルバッファ５９にロードする。圧縮テーブル２８のデータは、必ずしも全て利用されるものではないが、少なくともマスクデータ２８ａと、アドレステーブル２８ｂの１個あるいは２個のアドレス（タグ）の値は参照される。したがって、各照合器５０において共有資源であるＳＤＲＡＭ１１に対し、２～３回のリードをおこなうよりは、バーストリードにより１回だけＳＤＲＡＭ１１からリードし、照合器内部で必要値を読み出すほうが効率的である。上述したように、ＳＤＲＡＭからバーストリードするには時

間がかかるため、このテーブルロードフェーズからマスクデータ読み出しフェーズは、比較フェーズと並列に実行される。詳しくは後述するが、ＳＤＲＡＭ１１との間にデータキャッシュを置いた場合、照合器５０に内蔵のテーブルバッファは不要であるが、その場合でも、共有資源たるデータキャッシュへのアクセスを複数回おこなうよりも、やはりローカルにコピーしてアクセスするほうが効率的な場合がありえる。

【０１０１】

図２１に、比較ユニット５２をさらに詳しく示してある。本発明の照合装置５０においては、部分被検索キー３６のビット数が固定なので１テーブルのエントリ数は固定であり、また、圧縮テーブルなのでエントリとビットパターンの対応も固定である。したがって、部分被検索キー３６の値と各エントリとの比較は固定論理回路により構成可能である。本例の照合装置５０は４ビットの比較を行う。このため、１６個の比較器ＣＭＰ０からＣＭＰ１５を備えており、比較器ＣＭＰ０（エントリＮｏ．０）は入力被検索キー３６と値０との比較、比較器ＣＭＰ１（エントリＮｏ．１）は入力被検索キー３６と値１との比較、そして、比較器ＣＭＰ１５（エントリＮｏ．１５）は入力被検索キー３６と値１５との比較というように各比較器ＣＭＰｉで、入力値ｘと固有値ｉとの比較をおこなう。つまり、比較器ＣＭＰｉ（ｉは０～１５）はそれぞれのエントリに固有の論理回路である。

【０１０２】

比較ユニット５２は、比較器の代わりにルックアップテーブルＬＵＴを用いたハードウェアなどの他のハードウェアで構成することも可能であり、本例に限定されるものではない。

【０１０３】

図２２に示すように、各比較器ＣＭＰｉは入力値ｘとして被検索キー４ビットを取り、ＣＭＰｉ固有値ｉとの比較をおこない、ｘとｉとの比較結果「＜」「＝」「＞」の３値（２ビット）の比較結果（ＣＭＰｉ出力）を出力する。すなわち、各比較器ＣＭＰｉは前項のような論理に基づき結果出力するよう比較器毎に論理圧縮し最適化した回路で構成される。

【０１０４】

図２３は、比較ユニット５２の異なる例を示している。並列に実行されるテーブルロードフェーズ～マスク読み出しフェーズに時間がかかることを考えれば、必ずしも上記のように１６個の比較器ＣＭＰｉを並列する必要はない場合がある。図２３は、１個の比較器ＣＭＰｘを１６回繰り返し利用する構成であり、処理時間のバランスが取れるのであればハードウェアを削減できる。もちろん、テーブルロードフェーズ～マスク読み出しフェーズの所要時間との兼ね合いで比較器ＣＭＰｉの必要個数は１個に限定されることはなく、２個以上であっても良い。

【０１０５】

図２４に、判定ユニット５４のマスク器５４ａの構成を示してある。マスクフェーズではマスク器ＶＬＤ５４ａにより無効エントリのマスク処理がおこなわれる。各エントリをマスクするマスクエレメント５４ｅは各エントリに対し同一論理の回路である。マスクデータ２８ａは、実際のエントリ（部分エントリ）２６の有効無効を各１ビットのマスク値で表現したものであり、比較フェーズでの各エントリ（仮想的なエントリ）の比較結果から、マスクデータ２８ａにより無効エントリの比較結果を取り除き、実際に登録されている各エントリ２６との比較結果に変換する。

【０１０６】

各マスクエレメント５４ｅは、図２５に示すように、比較フェーズのＣＭＰｉ出力２ビットとマスク値１ビットの合計３ビットを入力とし、「×」「＜」「＝」「＞」の４値（２ビット）を出力とする。ここで、「×」は無効エントリ、「＜」「＝」「＞」は有効エントリである。

【０１０７】

図２６から２８に、判定ユニット５４の選択回路５４ｂの選択論理を示してある。マスクフェーズ後の各エントリのＶＬＤ出力は、「×」（無効）、「＜」（キー３６＜エントリ２６）、「＝」（キー３６＝エントリ２６）、「＞」（キー３６＝エントリ２６）の４状態であり、この選択フェーズでは、これらのＶＬＤ出力の中から完全一致検索、範囲一致検索の照合結果を得て、次段へ繋がるエントリを選択する。図２６に示した選択論理は、完全一致検索の選択論理であり、

マスク器ＶＬＤ出力が「０１」となるエントリを選択する。図２６に示した選択論理により、ＶＬＤ出力が「０１」のエントリのみの出力G_f（ｉ）（完全一致を示す出力）が「１」となる。したがって、出力G_f（ｉ）が１となったエントリＮｏを取得することにより、完全一致のエントリN_fを得ることができる。また、出力S_fは出力G_f（ｉ）が１となるエントリがあったかどうかを意味する。バイナリビットツリー方式ではエントリとしての重複登録はありえない。従って、G_f（ｉ）の１６出力のうち、完全一致エントリがあれば、唯一であることは保証される。出力G_f（ｉ）による「＝」の選択は範囲検索においても照合継続処理で使用する。

【０１０８】

図２７に右倒れ一致検索型の範囲検索の照合継続選択論理を示し、図２８に左倒れ一致検索型の範囲検索の照合継続選択論理を示してある。範囲検索の場合、照合継続は完全一致同様の「キー３６＝エントリ２６」の選択であり、照合終了の選択は、範囲検索を右倒れ一致型でおこなうのであれば「キー３６＞エントリ２６」となる最大エントリを選択する。左倒れ一致型であれば、「キー３６＜エントリ２６」となる最小エントリを選択する。したがって、右倒れ一致の場合はＶＬＤ出力が「１１」のエントリのみの中間出力F_r（ｉ）が「１」となり、左倒れ一致の場合はＶＬＤ出力が「１０」のエントリのみが出力F_l（ｉ）が「１」となる。

【０１０９】

そして、範囲検索において、近似候補を示す出力G_r（ｉ）では、出力F_r（ｉ）が「１」のエントリのうち、最大値のエントリ（エントリ番号が最大のエントリ）のみが「１」となり、その近似候補のエントリＮｏを示す出力N_rが決定する。出力S_rは出力G_r（ｉ）が「１」となるエントリの有無を意味する。同様に左倒れ一致型の場合は、出力G_l（ｉ）では、出力F_l（ｉ）が「１」のエントリのうち、最小値のエントリ（エントリ番号が最小のエントリ）のみが「１」となり、近似候補となるエントリＮｏを示す出力N_lが決定する。出力S_lは出力G_l（ｉ）が「１」となるエントリの有無を意味する。

【０１１０】

図２９に、右倒れ一致型の場合の出力の変化を示してある。図２９（ａ）は比較フェーズ後のＣＭＰｉ出力であり、図２９（ｂ）は、テーブルより読み出したマスクデータ２８ａである。図２９（ｃ）はＶＬＤ出力を示し、ビット毎の（ａ）×（ｂ）となる。図２９（ｄ）はキー値＞エントリ値となるエントリ群の抽出F_r出力であり、図２９（ｅ）の中間出力H_rは、出力F_rから右エッジ（キー値＞エントリ値となる最大エントリ）を抽出したものである。図２９（ｆ）は、最終出力G_rであり、出力H_rを利用し、キー値＞エントリ値となる最大エントリのみとする。これを見てもG_r（ｉ）が「１」となるエントリ、すなわち近似候補となるエントリは１箇所であることは明らかである。一方、図２９（ｇ）は一致検索の出力G_fを示しており、ＶＬＤ出力をもとにキー値＝エントリ値（０１）となる一致エントリを抽出している。本発明の照合方法においては、テーブル２８への重複登録がない構造なので、この一致エントリの唯一性は自明である。出力G_rでもG_fでも、演算の結果、該当エントリは存在しないか唯一存在のいずれかとなる。

【０１１１】

図３０に、左倒れ一致型の場合の出力変化を示してある。右倒れ一致型の場合と同様にG_lの16出力のうち、G_l（ｉ）が「１」となる近似候補のエントリは唯一であることは保証される。

【０１１２】

出力ユニット５５においては、判定ユニット５４から得られる上記の照合結果に基づいて各パラメータを更新する。照合タグ更新部５４ｂにおいては、選択回路５４ｂにより得られるエントリＮｏ（出力N_f）に従い照合タグ３４を更新する。選択回路５４ｂより得られる出力S_fを見て、出力S_fが０であれば更新はおこなわない。照合継続となるエントリがなく候補が確定するからである。出力S_fが１であれば、照合器５０にロード済みの現段階の圧縮テーブル２８のアドレステーブル２８ｂから、エントリN_fに対応するタグ情報（アドレス）を読み出し、照合タグ３４を更新する。圧縮テーブル２８においては、ワード位置＋０がマスクデータ２８ａであり、ワード位置＋１からにタグ情報２８ｂが記録されている。

【０１１３】

候補タグ更新部５５ｃにおいては、選択回路５４ｂにより得られるエントリＮｏ（出力N_r）に従い候補タグ３５を更新する。候補更新情報である出力S_rを見て、出力S_rが０であれば更新はおこなわない。新規候補となるエントリがないからである。候補更新情報S_rが１であれば、照合器５０にロード済みの現段階の圧縮テーブルから、エントリN_rに対応するタグ情報を読み出し、候補タグ３５を更新する。ここで候補タグ情報３５には、タグ情報だけでなく、照合段階を示す候補段階情報も記録することが有効である。この候補段階情報は、候補タグに対応するテーブル段位置に設定する。つまり、候補タグがどのテーブル段であるかを示す情報である。この候補段階情報は、パイプラインが照合段数でフィックスされている分類装置１５においては、候補確定後のバイパス処理で必要となるものである。

【０１１４】

タグアドレスからテーブル段位置が一意に決定できるようにテーブル２８が配置されている場合は、候補段階情報は候補タグ情報に含まれるため、別処理は不要である。そうではない場合、照合器５０は自分がどのテーブル段数の処理を担当するのかの情報を保持し、その情報に従い、次段の情報を候補段階情報として候補タグ情報３５の中にセットする。照合器５０をパイプライン利用する場合、テーブル段位置とはつまり照合器ＩＤであるので、自器ＩＤが分かり、自器に対しパイプライン次段にあたる照合器ＩＤが分かればよい。パイプライン次段の照合器との関係は固定なので、照合器の初期化時に定めればよい。次段の照合器ＩＤを使用するよりは、テーブルのタグアドレスに照合段階ＩＤを含めてしまうほうが簡素である。データキャッシュを用い、データキャッシュのコントロールをおこなう場合、テーブルは段位置別にグループ化されたアドレスとなるので、それを流用できる。

【０１１５】

コマンド更新部５５ａは、選択回路５４ｂにより得られる照合継続情報である出力S_fに従い、次段に供給するコマンド３７を更新する。照合継続情報である出力S_fが０であれば、照合継続がなくなり候補が確定したことを意味するする

ので、「ＣＭＤ＝MATCHRDBP」を出力する。これにより、次段の照合装置５０では候補タグ３５と一致している照合タグ情報３４によるバイパス処理が行われ最終結果が出力される。照合継続情報S_fが１であれば、照合継続であるので、「ＣＭＤ＝MATCHEXEC」を出力する。これにより、次段の照合器装置でも照合タグ３４により継続して照合が行われる。

【０１１６】

出力ユニット５５は、さらに、コマンド更新部５５ａにより決定した次のコマンド３７の値に従い、出力先を決定する。次コマンド３７が「ＣＭＤ＝MATCHEXEC」であれば、照合は継続するので次段の照合装置５５にパケット化された照合継続情報３３を出力する。次コマンド３７が「CMD=MATCHFIN」であれば、バイパス読み出し後の出力処理であるから、出力をスルーとして最終段小照合器の次（出力器）に送り出す。スルー出力が照合器パイプラインを順次通過しながら出力されるのであれば、次の照合器５０に最終出力を送り出す。バス３９を使ってショートカットするのであれば、バス３９に最終出力を出力する。

【０１１７】

次コマンド３７が「ＣＭＤ＝MATCHRDBP」であれば、照合は終了し候補が確定するので、候補タグ情報３５によるバイパス処理をおこない、検索結果を決定する。この際、候補タグ情報３５は必ずしも次段テーブルを指しているわけではなく、それ以前の段を指している場合がありうる。したがって、次段の照合装置５０に出力できるとは限らず、候補段階情報により候補タグを扱える段階の照合装置５０を特定し、照合継続情報３３を出力する。

【０１１８】

図３１にコマンド３７の一覧を示してある。なお、これは完全一致検索、１次元範囲検索に加え、後述する登録・削除も含めたものである。

バイパス処理とショートカット処理

図３２に、図１６に示した分類装置１５においてバックトラックのあるバイパス処理を実行している様子を示してある。この分類装置１５においては、バイパス用のバス３９を備えており、バス３９に照合継続情報３３を出力することによ

りバックトラックを発生させている。図１６に示した分類装置１５はメモリアクセスタイムを改善するためにメモリ分離型にしているために、あるタグアドレスの指すテーブル２８は、それに対応する段階の照合装置５０からしかアクセスできない。ところが、バイパス処理移行（候補確定）時に得られる候補タグ情報３５は、照合器パイプライン上の次段とは無関係に、自器より前のいずれかの照合器５０に属するものである。従って、バイパスバス３９により候補タグ３５に従って処理パイプラインを逆行できるようにしている。

【０１１９】

図３３に、図１５に示したメモリ共用型の照合器パイプラインを備えた分類装置１５においてバイパス処理を実行する様子を示してある。メモリ共用型の照合器パイプラインを備えた分類装置１５の場合、全ての照合器５０がメモリ１１に格納された任意のテーブル２８にアクセスできるので、バックトラック用のバイパスバスは必要ない。メモリ共用型の照合器パイプラインでは、次段の照合器５０がバイパス処理をおこなう。したがって、最終段での次段のバイパス処理発生に対応するため、最終段の照合器５０の次にバイパス専用のバイパス処理器５８を設ける。照合器Ｆで照合失敗、すなわち、照合器Ａ～Ｆのうち最後に更新された候補が確定することにより次段で候補タグバイパス処理が発生した場合、次の照合器Ｇがバイパス処理をおこなう。照合器Ｇは本来、ある段のテーブル２８の処理パイプラインを構成しているが、メモリ共用型のため、任意の段のテーブル２８にアクセスできる。従って、照合器Ｆで照合失敗し、候補確定が発生し、照合器Ｇにてバイパス処理をおこなう場合、テーブルＢ～Ｇのどの位置の候補であっても、そのテーブルにアクセス可能である。

【０１２０】

バイパス処理が完了し結果タグ情報（最終照合結果）が得られて以降の照合器５０は、結果タグ情報のスルーのみを行う。最終段の照合器Ｇでは、照合継続が検索成功（完全一致エントリ）を意味し、照合失敗がテーブルＢ～Ｇあるいは結果タグのいずれかを指す候補タグが得られる事を意味する。照合器Ｇで照合継続により検索成功の場合、照合器５０から出力されるタグ情報は結果タグ情報であり、それを受けた最終段バイパス処理器５８では、検索終了としてステータスだ

けを更新し、バイパス処理は不要である。

【０１２１】

一方、照合器Ｇで照合失敗により候補確定の場合、候補タグ情報３５がテーブルＢ～Ｇのいずれかを指すタグ情報である可能性がある。したがって、バイパス処理器５８はバイパス処理を行い、候補タグ情報３５の指すエントリからテーブル２８にアクセスしてバイパス値（ＢＰ）を読み取り、検索終了ステータスにする。照合器Ｇで候補確定であり、候補タグ情報が照合器Ｇで更新された場合は、テーブルＧに記されたタグが結果タグ情報となるため、更新された候補タグ情報は結果タグ情報、すなわち最終照合結果となっている。従って、バイパス処理器５８ではステータス＝検索完了にするだけで、バイパス処理は不要である。

【０１２２】

以上まとめると、バイパス処理器５８は入力が結果タグ情報であればスルーし、そうでない場合は、テーブルＢ～Ｇのいずれかを指すタグ情報が得られるので、バイパス処理をおこなう。結果タグ情報か否かの識別は、タグアドレスの上位ビットを識別用の情報として使うか、パイプラインを流れるデータに情報を付与することで行うことができる。

【０１２３】

図１７に示した繰り返し型の分類装置１５の場合は、全ての照合器が任意のテーブル２８にアクセスできるので上記と同様にバイパス用のバスは必要ない。さらに、繰り返し型の場合、制御用のカウンタ値をバイパス処理を意味する値にし、繰り返し処理の一環として自器にてバイパス処理をおこなうことができるので、バイパス処理器５８も不要である。

【０１２４】

ところで、バイパス処理にて得られるタグ情報は最終結果であり、バイパス処理以降の照合器パイプラインはスルーしてパイプライン最終段まで進み分類装置１５の出力となる。また、完全一致検索での照合不一致（検索失敗）あるいは範囲検索での照合不一致または候補なし（検索失敗）の場合も同じく、検索失敗のステータスを持って自器以降の照合器パイプラインをスルーしていく。バイパス処理のためのバックトラックがなければ、分類装置１５において、被検索キー２

１を入力した順番と、それの照合結果が出力される順番とが保障されるが、バイパス処理のためのバックトラックを許容するので、入力の順番と出力の順番が同一であることは保障されない。このため、被検索キー２１にＩＤ３２をセットにして供給するようにしている。したがって、スルーする場合も、図３４に示すように、照合器パイプラインをスルーで進んでいく必要はなく、図３５に示すように、バイパス用バス３９を転用して照合器パイプラインをショートカットすることが可能となる。

【０１２５】

さらには、ショートカット先は常に最終出力部なので、図３６に示すように、バイパス用バス３９とは別にショートカット用のバス３８を設けてもよい。バイパス発生とほぼ同数ショートカットは発生するため、常に決まった宛先のためにバイパス用バス負荷を高める必要はない。バイパス用バス３９のバイパス処理での利用率が５０％以下であるなら、ショートカットもバイパス用バス３９を使う方がハードウェアが増大しないメリットがある。

【０１２６】

メモリ共用型照合器パイプラインを備えた分類装置１５においては、バイパス用バスは不要である。このため、図３７に示すように、ショートカット用のバス３８を新規に設けることが望ましい。繰り返し型の分類装置の場合は、制御用カウンタ値の判定でバイパス処理後は繰り返し終了としておけばよく、ショートカット経路は不要である。

【０１２７】

検索テーブルの管理（更新処理）

以下では、照合装置５０のテーブル更新ユニット６０においてエントリ２２を追加あるいは削除する処理について説明する。まず、上述したように、圧縮テーブル２８はマスクデータ２８ａとアドレステーブル２８ｂで構成される。したがって、マスクデータ２８ａにビットあるいはフラグで示された部分エントリを追加したり削除したりするだけで基本的には検索テーブル２８の更新が可能であり、ソーティングや正規化などの複雑な処理は必要ない。このため、検索テーブルの管理は非常に簡略化され、高速に処理される。したがって、エントリの登録や

削除といったテーブル２８の管理を、被検索キー２１の照合処理と並列実行、つまり削除(登録)と検索がパイプライン並列に実行させることが可能となる。ただし、エントリを追加することにより検索テーブルに用意されるバイパス値ＢＰを更新する必要があるケースがあり、その判断に若干の処理が必要になる。新しい検索テーブルを生成するための処理も必要になる。しかしながら、バイパス値ＢＰは、検索テーブル２８にいったん設定すると、照合処理において、近似候補が決定された後の処理をショートカットできるので、検索テーブルの更新時にバイパス値ＢＰを設定しておくことには意味がある。

【０１２８】

まず、テーブル２８の管理と検索がパイプライン並列に実行される状況を想定すると、照合中の整合性を保つための排他管理が要求される。検索（照合）においても、処理対象のテーブル２８を占有すれば整合性保持は容易になるが、ツリー上位テーブルにおけるシリアル化がボトルネックとなり、照合処理の並列実行を阻害しかねない。したがって、検索においては「排他は確認するが、テーブルは占有しない」アクセスにより検索の間の並列性を生かしつつ、登録・削除側、すなわちテーブル更新処理では「排他を確認し、さらに、テーブルを占有する」アクセスで整合性も維持する。なお、検索と登録（削除）の間では、先行する登録（削除）の占有に対し後続の検索が排他確認の待ちをおこなうことでシリアル化される。

【０１２９】

このため、検索は上記のように「排他は確認するがテーブルは占有しない」状態で並列に実行される。このため、登録・削除との関係では、先行する登録・削除に対しては排他管理により追い越しが発生しないようにする。後続する登録・削除については、検索の上位構造における並列性起因の問題であり、下位構造たる検索で対応できるものではないため対処しない。

【０１３０】

登録は「排他を確認し、テーブルを占有する」状態でシリアル化されテーブル単位で逐次実行される。また、各テーブルのバイパス情報を更新する必要があり、その情報は下位から上位へと伝播するものであるため、登録時に何らかの形で

下位から上位への伝播をおこなわねばならない。考えられる方式としては、上位で待機し下位からフィードバックする待ち合わせ方式と、上位から下位に下っていき双方向リンクにより下位から上位に戻る方式との２種ある。待ち合わせ方式の場合、何らかの待ち合わせ機構をハードウェア資源で実現しなければならず、その資源量により並列数やテーブル段数等が制約を受けるが、処理速度的には有利である。ハードウェアによる待ち合わせ機構ではなくメモリによる待ち合わせにすることも可能であるが、メモリアクセスが増えるため、処理速度面での有利性がなくなる可能性がある。双方向リンク方式の場合、資源量起因の制約はないものの、１テーブルのメモリ量増加やリンク読み出しによるメモリアクセス増加（処理速度に影響）など、メモリ周りに不利な点をかかえる。

【０１３１】

待ち合わせ方式による登録

図３８に、待ち合わせ方式による登録の概要を示してある。この方式では、テーブル化されたバイナリビットツリーを上から下に辿りながらバイパスＢＰの更新が確定したテーブル２８は待ちなし（Ｗなし）で更新処理をおこない、バイパスＢＰが未確定のテーブル２８は各段で待たせていく（Ｗあり）。新規のテーブル２８ｎを作成する必要がある段階に到達した場合は、新規のテーブル２８ｎを作成した後に、その段階以下の新規なテーブル２８ｎとの連結と、上位で待機中（Ｗあり）（バイパスＢＰの確定待ち）となっているテーブル２８へのフィードバックを並列におこなう。

【０１３２】

図３８において、テーブルＡでは追加するエントリ２２の部分エントリ２６は既存（エントリが既存）であり、追加（登録）されるエントリの部分エントリ２６がＭＡＸエントリ（最大のエントリ）ではない。このため、下位のテーブルに対するエントリの登録がどうなっても、このテーブルＡでのバイパスＢＰの値に影響を与えずバイパス値ＢＰを更新する必要はない。従って、待ち（Ｗ）なしでＢＰ更新も不要である。テーブルＢも追加するエントリ２２の部分エントリ２６は既存（エントリが既存）であるが、その部分エントリ２６はテーブルＢのＭＡＸエントリに該当する。従って、下位での結果次第でバイパス値ＢＰを更新する

必要がありえる。このため、下位が確定するまでは、このテーブルＢのバイパス値ＢＰの更新が必要かどうか確定しないため、下位の更新待ちで待機する（Ｗあり）。

【０１３３】

テーブルＣは追加するエントリ２２の部分エントリ２６が新規（エントリが新規）であり、その新規エントリ２６がＭＡＸエントリとなる。したがって、このテーブルではバイパス値ＢＰの更新が必要になる。この段階で、バイパス値ＢＰの更新が発生したので、上位のテーブル２８へ通知する。Ｗあり状態となっているテーブルＢでは下位からの通知を受けてバイパス値ＢＰを更新し、更に上位の待ちがあればそこへ通知する。この例では、テーブルＡはＷなしなので上位への通知は不要である。

【０１３４】

一方、テーブルＣから下への流れは上記の上位フィードバックと並列に進む。テーブルＣで次段が新規テーブル２８ｎであると判明するので、空テーブル管理キュー６９に要求し、応答としてテーブルＤになるべきテーブルのタグアドレスを受け取る。そして、テーブルＣのマスクデータ２８ａに新しい部分エントリ２６のフラグを追加し、アドレステーブル２８ｂに新しいエントリＴタグを書き込む。テーブルＤ以降のテーブルは常に新規なので、空テーブル管理キュー６９に対して新たなテーブルのアドレスを要求し、マスクデータ２８ａおよびアドレステーブル２８ｂにデータをの書き込む。同時に、バイパス値ＢＰも更新する。
をおこなっていく。

【０１３５】

図３９に分類装置１５における照合器パイプラインでみた、待ち合わせ方式でエントリ２２を登録する構成を示してある。エントリ２２の登録は上位から下位へ照合器パイプラインを進みながら、テーブルを辿っていく。バイパス値ＢＰの更新保留となったら、各照合装置５０に対応する待機管理ユニット６８に必要な情報を記録したパケットを投入し、次段以降に進む。次段へは、直接の上位段となる親テーブルでの待機有無の情報が伝えられる。バイパス値ＢＰが確定した照合装置５０においては、親テーブルで待機有りなら、上位フィードバック（ＦＢ

）を発行する。この上位ＦＢと並列に、下位へのエントリ登録のパイプライン処理はおこなわれていく。上位ＦＢは待機管理ユニット６８で待ち合わせたパケットと合流し、保留していたバイパス値ＢＰを更新する。保留解除によるバイパス値ＢＰを更新する処理とは並列に、必要であれば待機管理ユニット６８から更に上位保留へのフィードバックを行う。

【０１３６】

図４０に、待ち合わせ方式でエントリを登録する処理において、更新処理ユニット６０が動作したときの照合装置５０の制御的な構成を示してある。ＤＴＡＧは結果タグ（登録データＴＡＧ）を示し、ＴＴＡＧはテーブルタグを示し、ＰＷＦは直接の上位段（親テーブル）での待機有無を示すフラグを示す。コマンド「ＡＤＤ」は上位から下位への既存のエントリのタグを辿って伝播される。コマンド「ADDNEW」は途中段以降の新規テーブルを確保する。コマンド「ADDUPDT」は上位へのフィードでＢＰの更新を再開する。コマンド「ADDWCLR」は上位で待機している照合装置５０に対するバイパス値ＢＰが不変の通知であり、待機解除に繋がる。

【０１３７】

照合継続情報３３と同様の経路で更新用のデータパケットが照合装置５０に伝達される。そして、ロードユニット５３により照合タグ３４で指示されたアドレスのテーブル２８がテーブルバッファ５９にロードされる（ステップ１２０）。コマンドデコーダ５１はコマンド３７をチェックし（ステップ１２１）、コマンド３７は追加「ＡＤＤ」から開始される。まず、ステップ１２２で、マスクを読み出し、ステップ１２３で、キー値とマスク値により次段が新規かどうかを判断する。次段が新規であれば、ステップ１２４で、空きテーブルキュー６９に対してテーブル２８ｎを要求し、応答を待つ間に、ステップ１２５で、マスクデータ２８ａの更新処理をおこなう。マスクデータ２８ａの読み出し値と、追加する部分エントリ２６の値の論理和（ＯＲ）を演算し、追加エントリに該当するビット（フラグ）を１にする。

【０１３８】

テーブルキュー６９から空テーブルを確保した応答が得られると、ステップ１

２６でその空テーブル２８ｎのアドレスをアドレステーブル２８ｂに書き込み、追加処理を伝播するための出力ＴＴＡＧに空テーブル２８ｎのアドレスをセットする。さらに、ステップ１２７で、出力するコマンドＣＭＤ３７を「ADDNEW」にセットする。

【０１３９】

コマンド３７が「ADD」であり、次段が既存であった場合、ステップ１２８で自テーブル２８のアドレステーブル２８ｂから次段のテーブルアドレスを読み出し、出力ＴＴＡＧにセットする。ステップ１２９で出力するコマンドＣＭＤ３７は入力コマンドと同じく「ADD」のままである。

【０１４０】

入力コマンド３７が「ADDNEW」であった場合、その段のテーブル２８ｎは前段処理により確保された新規テーブルなので、ステップ１３０でテーブル初期化をおこなう。これはマスクデータ２８ａだけの処理でよく、マスク値として該当するエントリビットを１に、他を０にする。初期化以降は、コマンド３７が「ADD」で新規エントリであった場合と共通の処理である。エントリの登録を照合装置パイプラインに伝播する過程で、一度、コマンド３７が「ADDNEW」、すなわち新規エントリになれば、それ以降の照合装置５０に供給されるコマンド３７は常に「ADDNEW」であり、上記の処理が繰り返される。また、上位のテーブル２８にフィードバックされるバイパス値ＢＰを更新する処理については、コマンド３７が「ADDNEW」の場合は常におこなう。

【０１４１】

コマンド３７が「ADD」の場合、すなわち、現段階のテーブル２８は既存のテーブルである場合は、ステップ１３１で、エントリの位置によって判断する。コマンド３７が「ADD」で、該当（新規）のエントリ２６が新規（次段TBLが新規）のエントリ２２の部分エントリで、それが最大エントリであった場合（ＭＡＸエントリ更新）、バイパス値ＢＰは更新が確定する。コマンド３７が「ADD」で該当（新規）のエントリ２６が新規のエントリ２２の部分エントリで、最大エントリではない場合、バイパス値ＢＰは不変で確定する。コマンド３７が「ADD」で該当エントリ２６が既存で最大エントリではない場合、バイパス値ＢＰは不変で

確定する。バイパス値ＢＰが確定する場合、更新するのであればステップ１３２で更新し、ステップ１３３で出力ＰＦＷを０にセットする。一方、コマンド３７が「ADD」で該当エントリ２６が既存で最大エントリの場合、下位の状況によりバイパス値ＢＰの更新が必要かどうか確定するため、待機管理ユニット６８へパケットを登録し、バイパス値ＢＰ更新を保留する。更新を保留した場合、ステップ１３４で、出力ＰＷＦを１とする。

【０１４２】

更新不要、更新確定の場合、状態が定まったので、入力ＰＷＦを見、上位待機があるかどうかを判断する。上位待機があれば、１段上の待機管理ユニット６８へフィードバック出力する。更新確定なら、ステップ１３５でコマンド「ADDUPDT」、更新不要なら、ステップ１３６でコマンド「ADDWCLR」とし、上位の更新要不要を指示する。

【０１４３】

図４１に待機管理ユニット６８の動作をさらに詳しく示してある。待機管理ユニット６８は、待ち合わせ処理と、その後のバイパス値ＢＰ更新再開および上位フィードの並列化をおこなう。コマンドを見て「ADDUPDT」なら下位でバイパス値ＢＰが更新されたので、ステップ１３７で、バイパス値ＢＰの更新を再開する。そのために照合器５０のキューにパケットが投入される。コマンドが「ADDWCLR」なら下位でバイパス値ＢＰの更新がなかったので、発火したパケットは破棄される。ただし、次の上位フィード処理はおこなう。また、コマンドによる処理とはまったく独立に、ＰＷＦ（上位で待ちがあるかどうか）をみて、ＰＷＦが1なら上位へフィードする。そのとき、ＩＤ，ＣＭＤは下位から来たものをそのまま上位へフィードする。

【０１４４】

照合装置５０では、コマンド３７が「ADDUPDT」なら待機していたバイパス値ＢＰ更新処理を実行する。上位へのフィードは待機管理側がおこなうので、バイパス値ＢＰを更新した後は照合器５０ではパケットを破棄する。

【０１４５】

双方向リンク方式による登録

図４２に、双方向リンク方式による登録の概要を示してある。この方式でのバイパスへの対応は、上から下に辿りながらエントリ登録をおこない、その際にバイパス値ＢＰの更新が確定したテーブルでは更新処理をおこなう。バイパス値ＢＰの更新が未確定テーブルはそのまま下位へ移動し　待ち合わせ方式による登録、待ち合わせ処理のような待ちは設けない。新規テーブル作成段に到達後、その段以下の新規テーブルの連結と、双方向リンクを辿りながらの上位のテーブルに対してバイパス値ＢＰの確定待ちへのフィードバックを並列におこなう。

【０１４６】

図４２において、テーブルＡでは、エントリが既存であり、ＭＡＸエントリではないため、下位がどうなっても、このテーブルＡでのバイパス値ＢＰの更新はない。テーブルＢでは、エントリ既存であり、ＭＡＸエントリに該当する。従って、下位での結果次第でＢＰ更新がありえる。下位が確定するまではこのテーブルＢのバイパス値ＢＰの更新が必要かどうか確定しないため保留する。テーブルＣではエントリ新規であり、その新規エントリがＭＡＸエントリとなる。従ってこのテーブルＣでバイパス値ＢＰの更新が必要となる。バイパス値ＢＰの更新が発生したので上位へ通知する（双方向リストを利用し上位テーブルＢへ遡る）。遡ってきたテーブルＢでは下位からの通知を受けてバイパス値ＢＰを更新し、更に上位へ遡っていく。バイパス値ＢＰの更新の必要がなくなった段で遡りは打ち切りである。一方、テーブルＣから下への流れは、待ち合わせ方式による登録と同様に処理待機上位への遡りとは並列に進んでいく。

【０１４７】

図４３に分類装置１５における照合器パイプラインでみた、双方リンク方式でエントリ２２を登録する構成を示してある。双方リンクを辿って上位にフィードバックするので待機管理ユニットは設けられていない。上位フィードバック（ＦＢ）はこの図では専用経路であるが、バス効率に影響がなければ検索バイパス用バス３９を共用してもよい。双方向リンクによる上位遡りは検索時のバイパス処理（任意の上位へ接続）とは異なり、常に直接繋がる上位テーブル（パイプライン前段の小照合器）へのフィードバックなので、ここでは専用経路による接続とした。エントリ２２の登録は上位から下位へ照合器パイプラインを進みながら、

テーブルを辿っていく。バイパス値ＢＰの更新が確定した照合器５０から双方リンクにより上位フィードバック（ＦＢ）を発行する。この際、待ち合わせ方式と同様に、上位ＦＢとは並列に、下位テーブルへの辿り（新規テーブルの接続）はおこなっていく。

【０１４８】

この双方向リンク方式においても、並列処理は下位への流れとバイパス値ＢＰ確定後の上位へのフィードバックの２並列である。また、上位フィードバックは完全な並列ではないがある程度パイプライン並列化できる。フィードバックによるバイパス値ＢＰを更新する時に、すぐに上位テーブルリンクを読み出して、バイパス値ＢＰを更新する処理とは分岐して更に上位へのフィードバックを発行することができる。ただし、検索・検索バイパスとの関係で待ち行列数が増えるため、全体のパフォーマンスを検討した上で、上位フィードバックのパイプライン並列化の実装は検討すべきである。

【０１４９】

図４４に、双方向リンク方式でエントリを登録する処理において、更新処理ユニット６０が動作したときの照合装置５０の制御的な構成を示してある。ステップ１２０から１３２の処理は、待ち合わせ方式と共通するので説明を省略する。バイパス値ＢＰの更新をおこなうのは、コマンド３７が「ADDNEW」の場合と、コマンド３７が「ADD」で新規エントリがＭＡＸエントリの場合である。前者の場合は上位へのフィードバックは不要である。コマンド３７が「ADD」から「ADDNEW」と切り替わる段のテーブルでフィードバック済のためである。したがって、ステップ１４０でコマンドをチェックし、後者の場合は、ステップ１４１で上位リンクを辿り、ステップ１４２で上位テーブルへ「ADDUPDTコマンドパケット」によるフィードバックを出力する。また、「ADDUPDTパケット」が得られると、ステップ１４３でマスクデータを読み込み、該当エントリがＭＡＸエントリかどうかでバイパス値ＢＰの更新を判断する。上位から下位への処理でエントリは登録済であり「ADDUPDT」を受け取った時点では常に既存エントリである。バイパス値ＢＰの更新をおこなった場合、更に上位テーブルへのフィードバックをおこなう。最上位テーブルでは上位リンクがＮＵＬＬであり、その場合はフィードバ

ックは終了する。

【０１５０】

待ち合わせ方式による削除

削除も更新処理であり、「排他を確認し、テーブルを占有する」状態でシリアル化されたツリーを構成するテーブルを、テーブル単位で逐次実行される処理である。検索との関係も上記と同様である。削除の場合、登録同様に下位から上位へ伝播するバイパス値ＢＰの更新があり、この上位への遡りを登録同様に待ち合わせ管理機構でおこなうのが待ち合わせ方式による削除である。登録とは異なり削除の場合は上位から下位へのエントリ検索時点ではバイパス値ＢＰの更新を確定できるケースはない。下位テーブルでエントリが０個になった場合にそのテーブルを開放し、上位へ通知し、上位テーブルのエントリを削除し、必要であればバイパス値ＢＰを更新する処理の流れとなる。この処理は下位から上位へ伝播していく処理である。登録と比べるとバイパス値ＢＰの更新は同じであるが、テーブル削除の上位伝播処理があるため、削除のほうが複雑になる。

【０１５１】

図４５に示すように、待ち合わせ方式では、コマンド３７が削除「DEL」のときで上位から下位へ辿りながら各段で待機し、最下位から上位に向かってコマンド３７を「DELUPDT」としてバイパス値ＢＰを更新し、テーブルを削除した情報をフィードしていく。この基本的な流れは待ち合わせ方式の登録とほぼ同様である。削除の場合、下位でテーブルが空になり開放されると、上位でのエントリ削除とそれに伴うバイパス値ＢＰの更新が発生する。たとえば、テーブルが空になると最終の照合装置５０がパイプラインから切り離される。テーブルが空ではない場合は、下位でのバイパス値ＢＰの更新情報を上位へ反映するだけでよい。テーブルが空になった場合、下位でのテーブル削除が上位でのエントリ削除につながり、上位でのエントリ削除の結果、バイパス値ＢＰの更新のため新しいバイパス値ＢＰを与えるエントリの下位への問い合わせするバイパス値問い合わせが発生する。自テーブル内の情報では新しいバイパス値ＢＰを与えるエントリのバイパス値ＢＰは得られないからである。このコマンドが「DELUPDT」のときの下位へのバイパス値問い合わせ（ＢＰ問い合わせ）は、登録にはなく削除に特有の処

理である。

【０１５２】

下位へのＢＰ問い合わせは以下の流れで行われる。ステップ１５１でエントリ削除にともなうバイパス値ＢＰ更新の発生した段で下位へコマンド「DELRDBP」で問い合わせ、応答待ちのパケットを待機する。下位の照合器５０では、コマンド「DELRDBP」を受けると自テーブルのバイパス値ＢＰを読み出し、ステップ１５２で上位へコマンド「DELWRBP」を送ってフィードバックする。応答待ちとの間で発火し、コマンド「DELWRBP」の処理として問い合わせで得られた新しいバイパス値ＢＰを書き込む。さらにステップ１５３で、バイパス値ＢＰの更新情報を上位へコマンド「DELUPDT」でフィードバックする。

【０１５３】

図４６に、待ち合わせ方式でエントリを削除（コマンドが「ＤＥＬ」）する処理において、更新処理ユニット６０が動作したときの照合装置５０の制御的な構成を示してある。ます、ステップ１６０で、入力パラメータＴＴＡＧで示されるテーブルをロードし、マスクデータ２８ａを読み込む。ステップ１６１で、マスクデータ２８ａと削除対象のエントリ（部分エントリ）をチェックし、対象のエントリが未登録ならステップ１６２でコマンド３７を「DELERR」として削除結果を最終出力する。

【０１５４】

ステップ１６３で次段のタグ情報を読み取り、ステップ１６４で自テーブルが最下段であるか否かを判断する。自テーブルが最下段ではなかった場合は、ステップ１６５で、下位からのフィードバック待ち合わせ用のパケット（ＩＤ，ＷＩＤ＝１，ＴＴＡＧ（自テーブルＴＡＧ），ＫＥＹ（自テーブルエントリＫＥＹ）を含む）を待機管理ユニット６８に投入する。ステップ１６６で、次段の照合器５０へコマンド「DEL」（ＩＤ，ＴＴＡＧ＝ＮｅｘｔＴＡＧ，ＫＥＹ＝ＳＨＩＦＴを含む）を出力する。

【０１５５】

一方、自テーブルが最下段であった場合は、ステップ１６７で、エントリに対応するＮｅｘｔタグ、すなわち、結果タグを読み出し、削除結果として最終出力

する。バイパス値BPの更新も含めた削除完了前に応答が戻るのがまずいようであれば、結果タグは上位フィードバックに載せてフローさせ、最上位まで戻りきったところで最終出力する。次に、ステップ１６８でエントリを削除する。本発明においては、テーブル２８のマスクデータ２８ａのエントリを削除するだけでよい。エントリを削除した後のマスクデータ２８ａの積算（和集合）ＳＵＭ－ＭＡＳＫが０ならばフラグＣＥＦを１（テーブルが空エントリ）にセットし、ＳＵＭ－ＭＡＳＫが１ならばフラグＣＥＦを０（テーブルにまだエントリが残っている）にセットする。また、エントリ削除前のマスクデータ２８ａと、削除後のマスクデータ２８ａを比較し、フラグＮＢＰを設定する。削除したエントリがバイパス値ＢＰを与えるエントリであれば、フラグＮＢＰを１すなわちＢＰ更新有りにセットする。

【０１５６】

ステップ１６９でフラグＣＥＦが１である判断すると、自テーブルが空エントリになるので、ステップ１７０で自テーブルを空きテーブル管理キュー６９に返却し、パケットのフラグＮＢＰとＤＴＡＧをＤＣ（ドントケア）にセットする。

【０１５７】

一方、フラグＣＥＦが０の場合、ステップ１７１でフラグＮＢＰをチェックし、ステップ１７２で、残ったエントリの中から新しいバイパス値ＢＰを与えるエントリ（ＭＡＸエントリ）を選択し、ステップ１７３で、バイパス値ＢＰを新しいＢＰエントリ対応するタグ情報（最終結果情報（結果タグ））に更新する。そして、パケットのフラグＮＢＰ（バイパス有効）を１にセットし、ＤＴＡＧにバイパス値ＢＰ（新バイパスＴＡＧ値）をセットする。なお、ステップ１７１で自エントリがＢＰエントリでなかった場合、バイパス値ＢＰの更新処理は不要である。そして、ステップ１７４で、コマンド「DELUPDT」により上位フィードバックの発火用パケットを出力する。

【０１５８】

図４７に、コマンド３７が「DELUPDT」のときの照合装置５０における更新ユニット６０の処理を示してある。この処理は、コマンド「DEL」が上位から下位に流れたときに待ちに入った照合装置５０が、最下段からのコマンド「DELUPDT

」を受けたときに、それをトリガとして発火する。まず、ステップ１８０でテーブルをロードする。ステップ１８１で、下位からフィードバックされたフラグＣＥＦが０ならば下位テーブルは消えていないので、自テーブルのエントリ削除は不要である。フラグＣＥＦが０のときは、ステップ１８２で、下位からのフラグＮＢＰをチェックし、下位テーブルのバイパス値ＢＰが変更になっているときは、さらにステップ１８３で自テーブルのマスクデータ２８ａと削除対象のエントリに従い、自己のフラグＮＢＰをセットする。そして、自ＮＢＰが１、すなわち、自テーブルのエントリがＢＰエントリ（ＭＡＸエントリ）であれば、ステップ１８６でバイパス値ＢＰを更新する。下位のフラグＮＢＰが０であれば、ステップ１８４で自ＮＢＰも０にセットしてバイパス値ＢＰの更新は行わない。そして、ステップ１８７で自段が最上位であるかチェックし、最上位でなければ、ステップ１８８で上位の待ちへフィードバックする。

【０１５９】

一方、下位からのフィードバックされたフラグＣＥＦが１ならば下位テーブルが消えたので、ステップ１８９で、自テーブルのエントリ削除をおこなう。フラグＣＥＦが１のとき、下位テーブルが消えたのでＮＢＰおよびＤＴＡＧはＤＣである。自テーブルのエントリ削除にともなうバイパス値ＢＰを更新する。さらに、エントリ削除後に自テーブルが空（ＳＵＭ－ＭＡＳＫ＝０）ならば、ステップ１９０でフラグＣＥＦｏｕｔを１にセットし、ステップ１９１で自テーブルを開放し、ステップ１９２で最上位であるか確認後、ステップ１９３で上位の待ちへフィードバックする。ただし、自テーブルが最上位テーブル(ルートテーブル)である場合、テーブル削除はおこなわない。ルートテーブルは空でも削除しない。また、ルートに対して上位の待ちは存在しないためフィードバックは行われない。

【０１６０】

エントリ削除後に自テーブルが空ではなかった場合、すなわちフラグＣＥＦｏｕｔが０のときは、ステップ１９４で、新ＢＰエントリ(ＭＡＸエントリ）を選択し、ステップ１９５および１９６で新ＢＰエントリに対応する下位テーブルからのバイパス値ＢＰの読み出しをおこなう。これは照合器パイプラインの都合に

より、次小照合器５０によるバイパス値ＢＰ読み出しを待ち合わせ管理機構で待つ方式でおこなわれる。待機側は、待ち合わせパケット（ＩＤ，ＷＩＤ＝２，ＴＴＡＧ，ＫＥＹ）を投入し、下位への依頼側は、コマンド「DELRDBP」により発行する。ＭＡＸエントリでない場合は、ステップ１９７でルートであるか否かを確認し、ステップ１９８で上位へフィードバックする。

【０１６１】

図４８にコマンド「DELRDBP」と「DELWRBP」のときの処理を示してある。上位に対するコマンド「DELUPDT」からの指示で下位にコマンド「DELRBP」が送られてくる。コマンドが「DELRDBP」であれば、ステップ２０１でバイパス値ＢＰを読み出し、ステップ２０２でＤＴＡＧにセットする。そして、上位はバイパス値ＢＰ読み出しを待っているので,上位の待機管理６８に対して待ち合わせ発火パケット（ＩＤ，ＣＭＤ＝ＤＥＬＷＲＢＰ，ＤＴＡＧ＝ＢＰ，ＷＩＤ＝２）を投入する。

【０１６２】

自器のコマンド「DELUPDT」からの待機パケットと下位からのコマンド「DELRDBP」完了後のパケットにより発火するコマンド指示が「DELWRBP」である。下位からのＤＴＡＧにバイパス値ＢＰが入っているので、ステップ２０３では、自テーブルを表すＴＴＡＧに従いＤＴＡＧ値を書き込む。自テーブルにおいてバイパス値ＢＰが確定したので、ステップ２０５で、コマンド「DELUPDT」による上位フィードバッグを再開する。ただし、コマンド「DELUPDT」と同様に、ステップ２０４でルートテーブルであるか否かをチェックし、ルートの場合は上位がないため、上位フィードバックはおこなわない。

【０１６３】

双方向リンク方式による削除

この方式では、まず、コマンド「DEL」により上位から下位にエントリを完全一致型の検索で辿り、最下位まで降りていく点は変わりない。コマンド「DELUPDT」による最下位から上位へのバイパス値ＢＰの更新、テーブル削除情報のフィードは双方向リストによるフィードでおこない、待ち合わせ方式のような待機機構は用いない。したがって、図４５に示した待ち合わせ方式とほぼ同様のフロー

になるが、上位に対するコマンドは待機管理ユニットではなく照合装置５０に供給される。

【０１６４】

また、下位でのテーブル削除にともなう上位でのBP更新については、待ち合わせ方式の場合同様にコマンド「DELRDBP」および「DELWRBP」でおこなうが、この処理も待ち合わせではなく双方向リストによるフロー移動でおこなう。したがって、待ち合わせ方式であれば、自テーブルのＫＥＹ値（削除対象のエントリ）は待機パケットが保持していたため下位からのフィードバックはＫＥＹを気にしなかったが、双方向リスト方式ではコマンド「DELUPDT」による上位フィードごとに該当段のＫＥＹを再び使用できなければならない。従って、上位から下位へのコマンド「DEL」による移動でＫＥＹを消費してしまう(消してしまう)のではなく、ＫＥＹ値はＳＨＩＦＴによる参照位置移動とし、コマンド「DELPDT」のときには逆方向ＳＨＩＦＴで該当段ＫＥＹ値を得られるようにしておく。つまり、ＫＥＹ値は４ビット８段テーブルであれば、３２（４×８）ビットのＫＥＹフィールドを持ち、コマンド「DEL」では上位ビットから下位ビットへの参照位置移動、コマンド「DELUPDT」では下位ビットから上位ビットへの参照位置移動をおこなう。

【０１６５】

したがって、コマンド「DELUPDT」のときは、双方向リストの上位リンクにより上位テーブルタグは得られる。前述のようにＫＥＹは参照位置の移動により、常に該当段のキー値をアクセスできる構造である。コマンド「DELUPDT」のときの下位へのバイパス値ＢＰの問い合わせの際も同様に、上位リンクによる上位テーブルタグの取得、ＫＥＹ値の適切な参照位置移動がおこなえる。したがって、下位へのバイパス値の問い合わせは、双方向リンクを用いて待ち合わせ方式とほぼ同様に行われる。

【０１６６】

図４９に、双方向リンク方式でエントリを削除（コマンドが「ＤＥＬ」）する処理において、更新処理ユニット６０が動作したときの照合装置５０の制御的な構成を示してある。待機管理ユニットを用いない方式なので、図４６に示した制

御系統図からステップ１６５が除かれた以外はほぼ同じ制御で処理は行われる。ステップ１７４で、コマンド「DELUPDT」で上位フィードパケットを出力する際、送出先のタグアドレスＴＴＡＧは上位リンクにより上位テーブルを指定し、エントリＫＥＹは上位方向へ逆シフトし、テーブル段位置と整合させる。

【０１６７】

図５０に、コマンド３７が「DELUPDT」のときの照合装置５０における更新ユニット６０の双方向リンク方式の処理を示してある。この処理においても、待機管理ユニットを用いない点を除けば、図４７と同じ制御で処理は行われる。なお、フィードバックする際に、送出先のタグ情報ＴＴＡＧは自テーブルの上位リンクにより上位テーブルを指定、エントリＫＥＹは上位方向へ逆シフトしてテーブル段位置と整合させることは上記と同様である。

【０１６８】

図５１にコマンド「DELRDBP」と「DELWRBP」のときの双方向リンクによる処理を示してある。これらの処理においても、待機管理ユニットを用いない点を除けば、図４８と同じ制御で処理は行われる。

【０１６９】

上記のように、本発明においては、エントリの追加および削除に伴うテーブルの管理にソーティングや正規化などの処理が不要となるので、エントリの追加削除に伴う照合処理の遅延は従来に比べて非常に小さなものになる。範囲検索におけるエントリとルールの関係については、右倒れ近似および左倒れ近似の説明において詳述している。たとえば、既存エントリが［ａｉ，無限大］［ｂｉ＋１，無限大］｛ｉは複数｝の時、新エントリ［ａ´，無限大］［ｂ´＋１，無限大］を加えるとき（登録する前に）、［ａ´，無限大］のルールにはａ´でヒットするエントリの（複数の）ルールを加え、［ｂ’＋１，無限大］のルールにはｂ´＋１でヒットするエントリの（複数の）ルールを加える。既存エントリが［ａｉ，無限大］［ｂｉ＋１，無限大］｛ｉは複数｝の時、［ａｊ，無限大］［ｂｊ＋１，無限大］を削除するとき（削除した後でに）［ａｊ，ｂｊ］に含まれるエントリから該当ルールを除く。

【０１７０】

データキャッシュ付き分類装置

図５２に、データキャッシュ１７を備えた分類装置１５の概略構成を示してある。データキャッシュ１７は、照合器パイプライン５００を構成する各照合装置５０とメモリであるＳＤＲＡＭ１１の間に設置される。図５３に示すように、本例のデータキャッシュ１７はＮウェイ・セット・アソシアティブ・キャッシュであり（Ｎウェイは２ウェイや４ウェイが一般的である）、キャッシュラインサイズがＳＤＲＡＭ１１のバースト転送サイズ、すなわち、照合テーブル２８のサイズと一致するように構成されている。データキャッシュ１７のキャッシュライン１７ａを１テーブルサイズとすることで、分類装置１５におけるテーブルアクセスが効率的に行われる。

【０１７１】

また、データキャッシュ内にライン単位のメモリ排他管理機構を導入することで、キャッシュメモリ上あるいはＳＤＲＡＭ上のテーブルデータの排他利用が可能となる。そのためには、キャッシュのラインサイズ、ライン数、セット数を適切に構成し、その構成に合わせてテーブルを割り当てれば良く、データキャッシュヒット率を高めることができる。そして、キャッシュシステム１７を採用することにより、汎用のＳＤＲＡＭ１１を用いながらさらに高速な分類が可能となる。

【０１７２】

図５４に、データキャッシュ１７を検索種別および照合段階で排他的に利用する例を示してある。すなわち、本例のキャッシュシステムにおけるテーブル２８のオンキャッシュ率を高めるために、照合テーブル２８の各段（たとえば、テーブルＡ－１とＡ－２）を異なるキャッシュエントリに割り当て、また異種の照合（たとえば、テーブルＡ群とテーブルＢ群）に対してもそれぞれ独立したキャッシュエントリを割り当てる。検索種別は、上流の制御ユニットあるいはアプリケーションから与えられるルートアドレス３１により簡単に識別できる。また、照合段階は、パイプライン段数に相当するので、識別は容易である。そして、照合テーブルの新規登録においても、テーブルの確保が必要になった場合、テーブル段位置に対応するキャッシュエントリにマッピングされるＳＤＲＡＭ１１のアド

レスを持つ空きテーブルを割り当てていく。

【０１７３】

また、本発明で用いられる照合テーブル２８はツリー型テーブルの構造となるので、下位段でテーブル数が広がっていく。したがって、下位段テーブルには複数キャッシュエントリを割り当てておくことが望ましい。図５４においては、照合Ａ用のテーブルＡ群と照合Ｂ用のテーブルＢ群がＳＤＲＡＭ１１の上にある。照合Ａの第１段テーブルＡ－１はキャッシュエントリ０にキャッシングされるアドレスに確保される。照合Ｂの第１段テーブルＢ－１も同様にエントリ０になるアドレスである。これらはキャッシュエントリが複数セットで多重化されているため、同時にオンキャッシュとなりうる。

【０１７４】

同様に照合Ａの第２段テーブルＡ－１はキャッシュエントリ１、照合ＢのＢ－２にはエントリ２が対応する。この場合、複数の照合がキャッシュエントリで多重化されず、それぞれ独立したキャッシュエントリが割り当てられている。キャッシュが４セットであれば、Ａ－２であれば全１６テーブルでキャッシュエントリ１の４セットを所有し、最新４テーブルがオンキャッシュである。下位段、特に、最終段のテーブルＡ－８およびＢ－８は数が多くなるので、１段に対して複数エントリを割り当てることも有効である。

【０１７５】

図５５に、空テーブル管理キュー６９が、メモリ１１の上でテーブル２８を管理する一例を示してある。検索種別毎にあるいはテーブル段毎にキャッシュエントリを分散配置させることでオンキャッシュ率を高めるためには、テーブル２８を管理する側、すなわち管理キュー６９が分類装置５０のテーブル確保要求に応じて適切なテーブルを割り当てる必要がある。空きテーブルキュー６９は空きテーブルを検索種別（レイヤー１）とそれに含まれるテーブル段位置（レイヤー２）の２階層に分類した個別キューで管理する。分類装置１５の初期化時に、キャッシュエントリ毎にテーブル２８をグループ化し、更にＲＯＭ等に設定された（あるいは外部からコンフィグレーションとして与えられる）（検索種別数×テーブル段数分の）空きテーブル個別キューとキャッシュエントリの対応に従い、テ

ーブルをキューにつないでいく。

【０１７６】

図５６に空きテーブルキュー管理機構６９の応答を示してある。空きテーブルキュー管理機構６９は、検索テーブルツリーへのエントリ登録時に照合装置５０からくるテーブル確保要求（ステップ２１１）に応じて、空きテーブルを取り出し応答する。この際、検索種別とテーブル段位置が必要となるため、要求情報としてそれらを要求パケットに持たせることとする。管理機構６９は、要求情報の検索種別（レイヤー１）・テーブル段位置（レイヤー２）に対応するキューから空きテーブルを１個取り出し（ステップ２１２）、応答としてそのアドレス（タグ情報）を照合装置５０に返す（ステップ２１３）。

【０１７７】

図１５に示したメモリ共有型の分類装置１５においてＳＤＲＡＭ１１と照合装置５０の間にデータキャッシュ１７がある場合、データキャッシュ１７の一つがミスヒットするとデータキャッシュ１７およびＳＤＲＡＭ１１はこのアクセスに占有されて他のアクセスは待たされる。図１６に示したメモリ分離型の分類装置１５においてもＳＤＲＡＭ１１と照合装置５０の間にデータキャッシュ１７がある場合も同様に、照合装置５０、データキャッシュ１７およびＳＤＲＡＭ１１はこのアクセスに占有される。これらはデータキャッシュ１７とＳＤＲＡＭ１１が同期的（シリーズ）に接続しているためである。そのため、１つ遅れると、その遅れは後段に影響する。パイプラインの効率を上げるには全てのステージにかかる時間を同一にしてやらねばならないが、ＳＤＲＡＭアクセスが入ると難しい。

【０１７８】

これに対し、要求応答方式のデータキャッシュ１７を採用することができる。すなわち、被検索キー２１に対して入力順に照合結果が出力されるというパケット順序の維持はバックトラックが発生することを前提とした段階で不可能であり、識別情報（ＩＤ）３２を導入することにより順序の問題はクリアしている。したがって、ある被検索キー２１の照合ためにテーブル２８が要求されたときに、オンキャッシュでなければ照合装置５０にその旨を通知する。照合装置５０は、処理を中断し、待ち行列に戻し、次の被検索キー２１のパケット３３を入れる。

データキャッシュ１７はＳＤＲＡＭ１１へアクセスする。一方、照合装置５０からテーブル２８を要求されたときに、オンキャッシュならば照合装置５０にテーブル２８を渡し、照合装置５０が処理を進める。さらに、照合装置５０において、照合タグ情報３４を先読みしてキャッシュシステム１７に渡し、必要なテーブル２８をプリロードするシステムは有効である。さらに、照合装置５０が照合のためにテーブル２８をバッファ５９にロードした時点で、次の照合タグ３４を入れてオンキャッシュを確認することも有効である。

【０１７９】

このように、本例のキャッシュシステム１７においては、検索種別および照合段階という要素でキャッシュラインの排他性を高めてオンキャッシュ率を高めている。従来のＣＰＵでのデータキャッシュにおいては、まず、キャッシュラインサイズをテーブルサイズに一致させるという発想はリニアアドレス空間を否定するので従来のキャッシュシステムからは得られないものである。さらに、照合テーブル各段を異なるキャッシュエントリに割り当てるという発想も、論理アドレス（のインデクス部分）とテーブルの割付決定を実行時に行うことであり、ＯＳの元ではＣＰＵで実行することは極めて難しい。ＣＰＵでは論理アドレスとテーブルの範囲割付決定をコンパイル時に行い、実行時には決められた範囲内から１テーブル分を確保するだけである。この実行時メモリ割付支援機構が認識するのは先頭論理アドレスとサイズである。動的メモリ割付は確保、開放を繰り返すとメモリに隙間を生じ、これをガベージコレクタで整理するというのが前提であった。従い、インデクス部分の衝突を避けて、キャッシュ効率を上げる仕組みを入れ込めなかった。

【０１８０】

これに対して、本発明における照合方法および照合装置においては、テーブルビット長を固定化（上記の例えでは４ビット）することによってテーブルを標準化したため、テーブルサイズとキャッシュラインサイズとを一致させることが可能になった。そして、キャッシュメモリ１７やＳＤＲＡＭ１１に対して検索種別および照合段階で排他的なエリアを割り付けたとしても、テーブル同士は照合タグ情報によりリストでつながれているので、エントリの（実行時の）削除、登録

に対応することができ、オンキャッシュ率を高めるために排他的な領域設定をすることが他の処理において障害となっていないといえる。

【０１８１】

履歴キャッシュ付き分類装置

図５７に、履歴キャッシュ１８を備えた分類装置１５の概略構成を示してある。照合装置５０の前に履歴キャッシュ１８をおくことで、検索処理をさらに高速化でき、検索処理の遅延時間を短縮できる。説明してきたように、本発明の照合方法および照合装置、さらに、分類装置および検索装置は、簡易なハードウェアでビット長の増加にも対応でき、照合速度も早く、さらに、汎用メモリを利用して十分な処理速度が得られるといった多くのメリットがある。しかしながら、複数の照合装置をパイプライン状に接続するか、あるいは照合装置を繰り返して使用することにより照合結果を導くので、同一の被検索ビットストリングが高頻度で表れるようなジョブにおいては、ＣＡＭあるいはＴＣＡＭと比べたときにレイテンシの面で優位とは言えない。したがって、レイテンシが重要となる検索においては、履歴キャッシュ１８を用いるメリットがある。

【０１８２】

図５７に示した履歴キャッシュシステム１８は、ＳＤＲＡＭ共用型履歴キャッシュシステムであり、照合テーブル２８の置かれるＳＤＲＡＭ１１を共用する構成である。図５８に示すように、キャッシングアルゴリズムとして８ビット化ハッシュ＋ＬＲＵを使用することができる。そのため、本例のキャッシュシステム１８は、ハッシュテーブル検索器１８ａと、ハッシュ用のキー一括比較器１８ｂとを備えている。ハッシュテーブル１８ｃは照合装置５０と同様に４ビット（１６エントリ）×２段のツリー構造で構成し、照合装置５０と履歴キャッシュ１８の間でデータキャッシュ利用方法の整合性を取るようにしている。

【０１８３】

履歴ＳＲＡＭキャッシュ付き分類装置

図５９に、履歴キャッシュ専用のＳＲＡＭ１３を備えた分類装置１５の概略構成を示してある。履歴キャッシュ１８のメモリを専用化した構成であり、テーブル２８を記憶したＳＤＲＡＭ１１と分離することにより、照合装置５０との競合

を防止して照合速度の向上を図ると共に、ＺＢＬ－ＲＡＭやＺＢＴ－ＲＡＭ等の応答性のよい低遅延ＳＲＡＭを履歴キャッシュ用に占有して履歴キャッシュ１８の性能を向上する。キャッシングアルゴリズムは８ビット化ハッシュ＋ＬＲＵを使用できるが、共用型の場合は照合テーブルとの整合性のあるメモリブロック構成が望ましかったのに対し、図６０に示すように、専用ＳＲＡＭ型の場合は、メモリビット幅やエントリの構成を履歴キャッシュに合わせて最適化することができる。

【０１８４】

論理演算装置（ＡＮＤ化器）の構成

上述した検索装置１０あるいは分類装置１５は論理演算装置（ＡＮＤ化器）１９を備えている。本発明に係る分類装置１５は、汎用型なので、ルートタグ３１により検索種別を上流の制御装置あるいはアプリケーションによって設定することができる。したがって、１つまたは複数の分類装置１５による検索種別の異なる検索結果（１次元範囲の検索結果）を複数用いて、ＡＮＤ化器１９で論理積を演算することにより、多次元範囲の検索を行うことができる。本例のＡＮＤ化器１９においては、１次元範囲の検索結果を集合として取り扱い、行列演算により多次元範囲の検索結果を単純な操作で得られるようにしている。

【０１８５】

まず、集合Ａが要素｛Ａ１，Ａ２，・・・，Ａｎ｝で構成されるとする。要素Ａｉがバイナリ（０／１）のｍｂｉｔ列“ａｉ１，ａｉ２，・・・，ａｉｍ”で表せるとする。同様に集合Ｂが要素｛Ｂ１，Ｂ２，・・・，Ｂｎ｝で構成されるとする。要素Ｂｉがバイナリ（０／１）のｍｂｉｔ列“ｂ１ｉ，ｂ２ｉ，・・・，ｂｍｉ”で表せるとする。

【０１８６】

このとき、集合Ａ，Ｂの積（ＡＮＤ）集合Ａ＆Ｂは、行列におけるＡ，Ｂの積Ａ×Ｂで表せる。詳しい演算式は、図６１に示す通りである。図６２に、集合Ａが２要素、集合Ｂが３要素、さらに各要素が３ビットで現れる例を具体的に示してある。左辺行列Ａは集合Ａの要素Ａ１，Ａ２が各行に並べられている。また、行列Ｂは集合Ｂの要素Ｂ１，Ｂ２，Ｂ３が各列に並べられている。右辺行列Ｚの

１行１列は「ａ１１＊ｂ１１＋ａ１２＊ｂ２１＋ａ１３＊ｂ３１」であり、「Ａ１×Ｂ１」である。これはＡＮＤ化器１９におけるブール演算の定義により、Ａ１とＢ１のビット比較結果（１＝一致／０＝不一致）を意味する。同様に行列Ｚを列方向で見ると、１列目は「Ａ１とＢ１の比較結果」と「Ａ２とＢ１の比較結果」になっている。同様に行列Ｚ２列目は「Ａ１とＢ２」と「Ａ２とＢ２」の比較結果である。つまり、行列Ｚの各列は行列Ａ全体（集合Ａ）と行列Ｂを構成する各列（集合Ｂ要素Ｂｉ）の比較結果である。

【０１８７】

したがって、行列演算の結果、例えば、図６３に示すように、右辺行列Ｚがなっていれば、左辺行列Ｘの｜ａ１１　ａ１２　ａ１３｜行（集合要素Ａ１）が集合Ａと集合Ｂの積集合Ａ＆Ｂである。つまり、行列Ｚの各行において行内に１つでも１が立つ行に対応する行列Ｘの行が積集合である。ここで「１つでも１が立つ行」を言い換えれば、行列Ｚの各行において行内の値の論理和ＯＲを取った結果が１となる行に対応する行列Ａの行が積集合である。

【０１８８】

したがって、行列Ａに対して、論理積を求める要素を掛け算し、１つでも１が立つ行が残れば、それらの要素の積集合が得られたことになる。図６４にＬＳＩ化を念頭に行列演算を一般化したものを示してある。左辺の行列Ｘは行毎に集合Ａの要素Ａ１，Ａ２，・・・である。左辺の行列Ｙは列毎に集合Ｂの要素Ｂ１，Ｂ２，・・・に加え、集合Ｃの要素Ｃｉ，・・・で構成される。列の数は限定されない。行列Ｚを列毎にみると、行列Ｘ全体と行列Ｙのある１列との演算となっている。行列Ｚは最終的には行毎に行内で論理和ＯＲが演算され、Ａ＊Ｂ＊Ｃ＊・・・の積集合要素が決定される。従って、行列Ｚはその値全体を保持しておく必要はなく、列毎にブール行列演算した上で、累積結果としてＯＲ演算していけばよい。

【０１８９】

図６５に、ＡＮＤ化器１９のハードウェア構成の一例を示してある。ＡＮＤ化器１９は、複数のブール行列演算レジスタ１９ａを有しており、集合Ａの各要素が各レジスタ１９ａにセットされる。さらに、各々のレジスタ１９ａに対応して

、レジスタ１９ａの演算結果の累積論理和（ＯＲ）を演算して結果を蓄積するアキュムレータ１９ｂを備えている。各アキュムレータ１９ｂは０に初期化される。このＡＮＤ化器１９に対し、入力要素１９ｉｎとして、集合Ｂの要素Ｂｉ、集合Ｃの要素Ｃｉなどの各列を順次に投入すると、入力列毎に以下の演算処理がおこなわれる。まず、ブール行列演算レジスタ１９ａと入力列による、ブール行列演算Ａｎ×Ｙｉ（ＥＮＯＲ演算、ＡＮＤ演算となる）が行われる。次に、ブール行列演算の結果とアキュムレータ値によるＯＲ演算が行われる。そして、ＯＲ演算の結果によりアキュムレータ１９ｂの値が更新され、最終入力後にアキュムレータ１９ｂに１が残る行に対応する集合Ａの要素群が集合Ａ，Ｂ，Ｃ，･･･の積（AND）集合、すなわち、Ａ＆Ｂ＆Ｃ＆･･である。

【０１９０】

ブール行列演算はＥＮＯＲ論理によるビット比較をＡＮＤ論理で繋いだものであり、回路的に見れば、各ブール行列演算レジスタはレジスタ値Ａｎと入力値Ｙｉとのｍビット完全一致比較結果を出力し、その比較結果をアキュムレータで累積ＯＲしていることになる。ブール行列演算レジスタの数やビット長にもよるが、ＥＮＯＲ論理およびＡＮＤ論理を組み合わせて回路を作る事は極めて簡単である。そして、入力値１個とレジスタ値複数個が一度に比較できるので、レジスタ数分の並列性がある。したがって、本例のＡＮＤ化器１９は、単純な構成でありながら、短時間で多次元範囲検索の結果を導出することができる。あるいは、並列レジスタがコスト高という場合は、レジスタを１個あるいは数を減らして、レジスタ値はメモリにし、入力値に対して１行分ずつ繰り返しブール行列演算することによっても本例のＡＮＤ化器１９は実現できる。

【０１９１】

レジスタ値をメモリに展開することにより、メモリの許す限り、ルールを入れることができるというメリットがあり、多数のルールあるいは要素を有する集合から論理積により要素を絞り込むのに適している。ただし、繰り返し処理でありメモリリードもあるため、処理時間は長くなる。ブール行列演算処理とメモリリードをパイプライン化することは可能であり、この構成によりメモリリード処理時間を隠蔽することも可能である。

【０１９２】

さらに、ＡＮＤ化器１９は内部にアキュムレータ１９ｂを持ち、またレジスタ１９ａに最初の集合の要素を保持する構造のため、複数の多次元範囲検索の並列実行を考えるとそれらは共有資源である。したがって、何らかの資源排他占有が求められる。一方、分類装置１５においては、複数検索の各次元の検索が並列におこなわれえるため、例えば、検索系S_1，S_2の各次元Ａ，Ｂについて、分類器からの出力はS_1Ａ，S_2Ｂ，S_2Ａ，S_1Ｂのように検索系も同検索内の次元も順序性のない出力となりうる。先に分類装置１５から出力された検索系がＡＮＤ化器１９を占有利用する方式でも、先に分類装置１５からの全次元出力がそろった検索系が占有利用する方式でも、いずれにせよ分類装置１５の出力が非同期・無順序であるため、ＡＮＤ化器１９の利用率および多次元範囲検索としてのスループットを高効率でおこなうためには、ＡＮＤ化器１９も並列化することが望ましい。ＡＮＤ化器１９のレジスタおよびアキュムレータをメモリ化した場合は、メモリ上での並列化でも効果が得られる。

【０１９３】

また、ＡＮＤ化器１９の動作を考えたとき、初期設定「BEGIN(レジスタ設定・アキュムレータ初期化)」と終了処理「END(アキュムレータからの読み出し)」が必要であり、同系内でも分類器入力と出力の間での順序性がないため、次元Ａだから初期設定処理、次元Ｂだから終了処理というように操作は固定的に決定できない。分類装置１５からの出力を検索系ごとにキューイングし、全次元の分類器出力が完了したのち、明示的に初期設定と終了処理をコントロールする方式となる。結局、これらの制御もあるので、分類装置からの出力がそろった検索系がＡＮＤ化器１９を排他占有する制御方式が望ましいといえる。

【０１９４】

ＡＮＤ化器１９の並列利用のためには、速度を要求されるならＡＮＤ化器１９そのものの複数用意することで対応できる。次善策としては、レジスタ１９ａおよびアキュムレータ１９ｂのメモリ保存によるメモリ上での並列化をおこなうことで対応できる。さらに、履歴キャッシュ１８を導入している場合、多次元範囲検索における履歴キャッシュ１８の使い方を工夫することで、ＡＮＤ化器１９の

使用率を下げることができ、ＡＮＤ化器１９を並列利用しなくても十分な処理速度を維持することも可能である。例えば、３２ビット５次元の多次元範囲検索であれば、１６０ビット（３２ビット×５）のキーによる履歴キャッシュ１８を採用し、履歴キャッシュ１８のエントリデータとして、初回検索時にＡＮＤ化器１９を使って演算した５次元の積集合結果を登録する。つまり、履歴キャッシュ１８には分類装置１５の最長一致検索の履歴、完全一致検索の履歴、１次元範囲一致検索の履歴をエントリデータとして登録することも可能であるが、多次元範囲検索の場合は、各次元個別の履歴ではなく、全体としての多次元範囲検索の履歴をエントリデータとして取り扱うことによりＡＮＤ化器１９による処理がクリティカルパスになることを抑制できる。

【０１９５】

上記に、幾つかの例を示しながら本発明についてさらに詳しく説明しているが、上記の例に本発明が限定されないことはもちろんである。たとえば、照合テーブル２８の数をさらに削減できる方法がある。照合装置５０を用いた検索においてはエントリビット列を４ビットずつの部分エントリに分割してテーブル化し、テーブル化ビットツリーにより登録し検索していく。この方法では、ある段以降が１エントリしか存在しなくてもテーブルを生成し、検索においてもテーブルを辿る必要がある。これを改善し、ある段以降が１エントリであれば下位テーブルを省略するテーブル構造を導入することが可能である。

【０１９６】

図６６に下位テーブルを省略する例を示してある。テーブルＢは本来のテーブル構成ではＢ－Ｃ－Ｄとなる１エントリが存在するが、下位が１エントリであるためＣ－Ｄを省略し、その代わりにテーブルＢに省略情報（エントリの省略した下位ビット列、省略段数、省略マークなど）を記録しておく。テーブルＥでは直接の下位接続はテーブルＦの１個であるが、エントリとしては孫テーブルで分岐したＧおよびＨの２エントリ以上が存在するため、省略はされない。

【０１９７】

テーブルＢの下位に収容される２エントリ目の登録時に、省略された下位テーブルは展開（生成）される。上図ではＢ－Ｃ－Ｘと繋がるべきエントリＸの登録

が行われる際、テーブルＢにて下位テーブルの省略情報を検出し、第一エントリに相当するテーブルＣ－Ｄの生成、第二エントリに相当するテーブルＸの生成をおこなう。

【０１９８】

ここで、第一エントリのテーブル生成と第二エントリのテーブル生成はシリーズにおこなうのではなく、テーブルＢの次の段は同ビット列なので共通テーブルＣを生成し、テーブルＣの次では異なるビット列なので分岐しテーブルＤおよびテーブルＸを生成するというように、２エントリが分岐する段までは統合的にエントリを生成することができる。したがって、エントリを登録する処理時間を短縮できる。

【０１９９】

また、分岐の次段では再度それぞれの下位が１エントリになる。このため、そこで省略テーブル形式でテーブル生成することになる。例えば、上図においてテーブルＤおよびＸが最下段ではなく、さらに下位に延びているのであれば、テーブルＣで分岐した次のテーブルＤおよびＸはその下位にそれぞれ１エントリしか存在しないため、下位テーブルは省略される。省略テーブル形式で生成されると、検索時は照合装置５０がテーブル２８の省略マークを見て、そのテーブル下位が省略されていることを検出し、次段のテーブルを読み出さずに最終照合結果を得ることができるので、照合時間も短縮できる。

【０２００】

圧縮テーブル（照合テーブル）２８には、４ビット×ｎ段分のエントリ省略ビット列が記録されているので、照合装置５０では、エントリビットをシフト読み出ししながら検索モードに応じた照合を行う。１エントリに対する完全一致・近似一致の照合であれば、上述した照合器５０の比較部５２を使う代わりに、専用化した比較器を実装したほうが効率的である。また、専用比較器を使用するのであれば、４ビット比較×ｎ回でおこなう必要はなく、リソースのバランスを見た上で多ビットの比較器とすることが望ましい。

【０２０１】

省略テーブルでは被検索キー２１と１エントリとの比較であるため、完全一致

検索であれば完全一致照合、範囲検索であれば、大・同・小が判定できる比較照合が可能な比較器でよい。また、比較後のエントリ選択処理は不要であり、比較結果から検索結果を判定するだけである。完全一致であれば比較で一致なら検索成功、不一致なら検索失敗。範囲一致であれば比較で大または一致であれば（右倒れ型）検索は成功、左倒れ型の場合も同様に比較が小または一致なら検索成功である。

【０２０２】

削除については、削除の結果、１エントリとなる部分を検出し、その下位からエントリビット列を集約し再度の省略テーブル化を行わなければならない。テーブル管理の処理時間は増加する可能性があるが、上述したように、照合時間を短縮できるので、メリットは大きい。各段のテーブル２８に収容エントリ数カウンタを設け、上位から下位へ削除エントリを辿る際にカウンタを減算し、１エントリのみとなるテーブル位置を検出する。そのテーブルを省略テーブルに変更し、その下位の省略対象エントリのテーブル列を辿り、最下段から上位へと省略ビット列（＝エントリＮｏ）を合成しながら上位へと上げていき、省略テーブルに集約して省略ビット列として記録するという方法がある。削除によるテーブル再省略化は、上述した更新処理よりも複雑でありテーブルアクセスも増える（時間がかかる）。従って、登録時のみ省略・展開をおこない、削除では再省略化をおこなわない方式もありうる。（検索頻度と登録・削除頻度とのバランスで、登録削除が一定以上多い場合、展開・省略の繰り返し、すなわち、テーブル確保・開放の繰り返しとなり、分類器全体の性能低下を招く可能性もある。

【０２０３】

図６７に、要求応答方式のメモリアクセスと小分類器内部のパイプライン化の概要について示している。上記においても説明したが、照合装置５０においてはマスクデータ２８ａの読み出しがボトルネックの１つである。マスクデータ２８ａがなければ、判定ユニット５４から下流の処理（ＶＬＤからＳＥＬおよび出力ユニット５５における処理）が実行できず、マスクデータ２８ａの読み出しと並列に実行できるのは主に比較ユニット５２の処理だけである。従って、照合器５０の内部処理もパイプライン構成とし、テーブルロード（マスク読み出し）フェ

ーズにてキャッシュ（ＳＤＲＡＭ）へメモリリード要求を出し、応答のあったものからＶＬＤ以降の処理をおこなうことが考えられる。そのために、ＶＬＤステージの前に待ち行列５７があり、そこで応答の待ち合わせがおこなわれる。

【０２０４】

図６７に示した例では、メモリ要求～応答の間にＣＭＰステージ５２を入れているため、待ち行列５７に待ち合わせ機構が必要となる。データキャッシュなしの場合は、メモリアクセスはＦＩＦＯであり待ち行列５７もＦＩＦＯによる待ち合わせでよいが、データキャッシュがあった場合、ヒット・ミスヒットにより順序性がなくなり、ＣＡＭ等のＦＩＦＯ型よりは高度な待ち合わせ機構が必要となる。データキャッシュなしの場合でも、メモリアクセスをともなわないモード（スルーなど）があるので、やはり単純なＦＩＦＯ待ち合わせ型の行列では無理がある可能性がある。メモリアクセスを伴わない場合も、ダミーのアクセス要求を出してダミー応答を戻させ、応答数とＦＩＦＯ待ち数を整合させればＦＩＦＯ型で対応できる可能がある。

【０２０５】

図６８に異なる照合装置５０の例を示してある。この例では、メモリアクセスを伴うものとそうではないものを分岐させ、メモリアクセスを伴うものは非順序の応答をＶＬＤ直前の合流待ち行列５７に戻す。一方、メモリアクセスを伴わないものは、待ち行列５７の前のステージ５７ｐで必要な処理があればおこない、合流待ち行列５７に入る。この場合、合流待ち行列５７は単なる２方向からの合流のみであり待ち合わせは不要となる。この構成では、待ち合わせ機構を不要とするため、メモリアクセスとＣＭＰ処理５２の（空間的な）並列化はなされないが、照合装置５０の内部のパイプラインにより時間的な並列化には有効である。

【０２０６】

図６９に、マスクデータを先読み対応可能なテーブル化によるメモリアクセスストールを解消する例について示してある。マスクデータの読み出しに係るボトルネックを解消するためにテーブル２８の構成を変更する。図６９（ａ）に示すテーブル２８は、そのテーブルのエントリ有無を示す情報としてマスクデータ２８ａがあり、次テーブルのアドレスを示すアドレステーブルを示すNextTAG×16

エントリで構成されている。図６９（ｂ）に示す改良型テーブル１は次段のアドレス「NextTAG×１６エントリ」２８ｂの次に、次段のマスクデータ「NextMask×１６エントリ」２８ｐを配置したテーブル構成となっている。また、図６９（ｃ）に示す改良型テーブル２は、次段のマスクデータ「NextTAG」２８ｂと次段のマスクデータ「NextMask」２８ｐのペアが１６エントリ登録されているテーブル構成である。いずれも、次段の照合テーブル２８のアドレスだけではなく、マスクデータ２８ａも備えているので、次段のマスクデータ２８ａをプリフェッチすることができる。

【０２０７】

ただし、改良型においても、テーブル管理や検索以外での利用のため、次段のマスクデータ「NextMASK」と同じ値が、次段のアドレス「NextTAG」の指すテーブル２８に記録されていることが望ましい。したがって、あるテーブルにはそのテーブル自身のマスクデータと、次段のマスクデータの２種のマスクデータが存在することが望ましい。テーブル下位からエントリを辿っていく場合に、上位のテーブルにしかマスクデータがない場合、そのテーブルだけでは隣接エントリへの移動が不可能になってしまうからである。

【０２０８】

次段のマスクデータをプリフェッチする構成にするのに伴い、照合装置５０の間をフローする照合継続情報のデータパケット３３にマスクデータ２８ａを追加する。この改良により、照合装置５０の処理においては、入力パラメータ（照合継続情報）３３としてその照合装置５０で使用する照合テーブルのマスクデータ２８ａが与えられる。このため、マスクデータ２８ａを必要とする判定ユニット５４から下流の処理と、ロードユニット５３の処理とを並列に実行することが可能となる。

【０２０９】

マスクデータ２８ａをリードすることに起因によるストールを解消するという点では、分類装置１５あるいは照合装置５０に内部ＳＲＡＭを設け、各テーブルのマスクデータ２８を照合装置５０の内部に配置することで並列度を向上することも可能である。

【０２１０】

以上に説明したように、本発明に係る照合方法および照合装置は、ＩＰヘッダのようなビットストリングを一致検索したり、範囲検索してルーティングするデータ管理装置１の検索機構１０、８ｓ、７ｓおよび６ｓとして好適なものである。しかしながら、本発明の照合方法および照合装置の適用範囲は、これらに限定されるものではない。たとえば、複数の照合装置５０を備えた分類器１５を様々に応用する場合、ＭＩＮ検索・ＭＡＸ検索があると便利である。バイパスをＭＡＸバイパスとしていた場合に、被検索キー｛１１・・・１１｝で右倒れ近似検索をおこなうことで、登録エントリの中からＭＡＸエントリを検索することはできる。だが、ＭＩＮエントリは検索できない。従ってこの場合、ＭＩＮ検索モードで検索することでＭＩＮエントリを取り出すことになる。ソートなどのように検索ではなく登録&取り出しで本発明に係る分類器１５を利用できる場面がある。このとき、ＭＩＮ検索およびＭＡＸ検索の両方を備えていることはメリットがある。

【０２１１】

照合器５０において、比較ユニット５２の処理を省略し、判定ユニット５４からなる検索モードが実行できるようにすることも有効である。たとえば、比較ユニット５２のＣＭＰｉからｈは常に「＝」（０１）が出力されるようにする。あるいはＭＩＮ検索・ＭＡＸ検索モードになったときには、比較ユニット５２をスルーして判定ユニット５４のＶＬＤ５４ａに「０１」が入力される回路としても良い。この検索モードにおいては、ＶＬＤ５４ａにより有効エントリのみ「０１」となり、ＳＥＬ部５４ｂにて、ＭＡＸ検索モードであれば比較結果が「０１」のエントリの中から最もエントリＮｏの大きなエントリが選択される。また、ＭＩＮ検索モードであれば最もエントリＮｏの小さいエントリが選択される。右倒れ・左倒れのG_l・G_r関連部でエントリ選択論理を「＞」（11）・「＜」(10)から「＝」（01）に変更することにより実現できる。テーブル乗り移り処理であるタグ更新部５５ｂは完全一致検索モードに準じ、出力S_r・S_lで指定されたエントリＮｏを次テーブルとした照合タグ情報３４の更新をおこなう。以上により、各段のテーブルで最小エントリ・最大エントリを辿っていくＭＩＮ検索・ＭＡＸ

検索機能を提供できる。

【０２１２】

次に、分類装置１５をソートへ適用する場合、ソート後のデータ読み出しのための拡張が必要となる。昇順先頭読み出し・降順先頭読み出しはＭＩＮ検索・ＭＡＸ検索で可能である。従って、昇順次読み出し・降順次読み出しのためのコマンドが必要となる。これを実現するためには、テーブルは双方向リンク型テーブルとする。また、昇順先頭読み出し（ＭＩＮ検索あるいは｛０・・・０｝近似検索）・降順先頭読み出し（ＭＡＸ検索あるいは｛１・・・１｝）の出力パラメータとして、最下段テーブルのＴＡＧ＋エントリＮｏを読み出しハンドラとして出力する。

【０２１３】

さらに、コマンド「ＮＥＸＴ・ＰＲＥＶ」を設け、読み出しハンドラ（最下段テーブルＴＡＧ＋エントリＮｏ）を入力パラメータとして受け取ると、次エントリの結果ＴＡＧと共に更新した読み出しハンドラ（最下段テーブルＴＡＧ+エントリＮｏ）を出力する。照合器５０の内部では、まず照合ＴＡＧ（読み出しハンドラの最下段テーブルＴＡＧ）３４に従い、テーブル２８よりマスクデータ２８ａを読み出す。読み出しハンドラのエントリＮｏに基づき、次エントリＮｏ（前エントリＮｏ）を算出し、テーブル２８から対応するエントリのアドレス「Next TAG」（最下段なので結果ＴＡＧ）を読み出す。もし、次エントリＮｏ（前エントリＮｏ）が別のテーブルになる場合は、双方向リンクの上位リンクを使い１段上位テーブルに遡り、そこで隣接エントリに移る。１段上位にも自エントリしかなかった場合は更に上位に遡っていく。上位テーブルで隣接エントリに移ったら再び最下段まで降りなければいけないが、これはＭＩＮ検索・ＭＡＸ検索モードによりおこなう。

【０２１４】

本発明の分類装置１５は、リレーショナルデータベース（ＲＤＢ）にも適用できる。バイナリビットツリーによる検索はデータベースではインデクスに相当する。したがって、複数のフィールドをバイナリビットツリー登録しておけば、これらをキーとしてデータベースを検索できる。検索の種類は最長一致でも、範囲

一致でも、完全一致でも対応できる。問題は複数のフィールドを検索後のＡＮＤ処理（積集合をとる処理）である。つまり「Ａ０＜身長＜Ａ１」と「Ｂ０＜体重＜Ｂ１」を別々に検索はできても「Ａ０＜身長＜Ａ１」かつ「Ｂ０＜体重＜Ｂ１」の集合をどう高速に得られるかが問題となる。予めフィールドが固定的に決定されているならば、そのフィールド群による複合（多次元）インデクスを生成しておくのが一般的である。対象フィールドが固定的に決定できず、複合インデクス（身長と体重の２次元インデクス）が生成できない場合、従来のＣＰＵ＋ＯＳの検索システムではおそらく次の方法で論理積を演算する。すなわち、レコード名をＲｉとすれば、「Ａ０＜身長＜Ａ１」の集合｛Ｒ１，・・・Ｒm｝（個数はm個）と、「Ｂ０＜体重＜Ｂ１」の集合｛Ｒ´１，・・・Ｒ´m｝（個数はm´個）を求め、二つの集合のメンバを総当りで比較し両方に属するメンバを残す。これがＡＮＤ集合になる。この場合、比較計算はm×m´回になる。

【０２１５】

これに対し、本発明の分類装置を用いると、次のような操作が可能である。すなわち、集合｛Ｒ１、・・・Ｒm｝を分類器１５に登録し、集合｛Ｒ’１，・・・Ｒ´m｝を通した時、前の集合で登録したメンバにヒットすればこれがＡＮＤ集合のメンバになる。この場合、比較計算はm＋m’回になり、処理速度が向上する。

【０２１６】

本発明の分類装置は、積集合（ＡＮＤ集合）だけでなく、和集合（ＯＲ集合）の演算も、ユニーク化も、また、ソートも簡単にできる。ＣＰＵ＋ＯＳの検索システムでデータベースにテンポラリーな印をつけて良いならばは次の方法で論理積（ＡＮＤ）を演算できる。集合｛Ｒ１，・・・Ｒm｝の各メンバを検索の都度印をつける。｛Ｒ´１，・・・Ｒ´m｝の各メンバを検索の都度前の印があるものをあげる。この場合、比較計算はm＋m’回になるが、不要な印の処理やディスク本体のデータベースへのアクセスなどの問題が起こり、時間がかかりすぎる。ビットマップ法は高速に論理積（ＡＮＤ化）できるがルール数に制約があるなど汎用化は難しい。

【０２１７】

これに対し、本発明の分類装置によるＡＮＤ集合をとる方法はパケットフィルタリング等の複数ルール集合のＡＮＤ集合化にも使える。本分類器１５では、例えば、５ＩＤ（５種類のビットストリームに対する検索）に対応する５インスタンスを並列に動かし、ＩＤ毎にヒットするルールに印をつけていく。既に４つの印を確認したインスタンスはこれをＡＮＤインスタンスへ通知する。ＡＮＤインスタンスは通知される都度、今の最優先ルールと比較し、更新する。５インスタンスが全て終了した後、今の最優先ルールに決定する。

【０２１８】

本発明の分類装置により、ツリー構造型の非リニアメモリアドレッシングの実現も可能となる。アドレスを検索キーとした形で、本発明の分類器をメモリ管理機構に取り込むことで、上位層からみて柔軟なアドレッシングが行えるメモリ空間が実現できる。これは従来のリニアアドレス空間メモリに対し、上位層が明示的に構造化管理をおこなっていた部分をメモリ管理機構が隠蔽カプセル化し、メモリを抽象化オブジェクト化することに他ならない。

【０２１９】

従来、ＣＰＵのコードでおこなっていたメモリのオブジェクト化アプローチとは異なりハードウェアベースのアプローチなので高速効率的にオブジェクト化が実現される。特に、検索のためのメモリ利用の場合、分類器によるアドレッシング（＝照合）が効果的に利用できる。

【０２２０】

図７０に、その一例を示してある。受信時、Ｉ／Ｆ２３０を経由してリニアアドレス（論理アドレス）の書き込みデータ（受信データ）を受け付けると、分類器１５は受信データをＳＤＲＡＭメモリ１１へ書き込む。分類器１５はアドレス変換機能２３２を持っているので、論理アドレス（リニアアドレス）から実アドレス（リストやツリーなど何らかの構造をもった非リニアアドレス）へ変換してＳＤＲＡＭ１１に書き込むが、Ｉ／Ｆ２３０を介してデータを送信している方は、リニアアドレスで書き込むことができる。送信時も同様である。Ｉ／Ｆ２３０からリニアアドレスで読み出すときに、分類器１５を経由することで、アドレス変換をおこない、リニアアドレスから実アドレスへ変換し、非リニアなアドレス

からの連続読み出しに対応できる。

【０２２１】

ＣＰＵ２３５も分類器１５によるアドレッシングを利用できる。ＣＰＵ２３５はデータフロー動作ではないため、分類器１５が提供するアドレッシング配下のメモリ１１だけではコード実行等に支障がある。従って、コード蓄積やデータ加工用として独自にメモリ２３６を持ち、分類器１５の配下のメモリ１１はパケットメモリとして利用する。ルータ等を考えた場合、ＣＰＵ２３５は分類器１５によるＴＣＰ／ＩＰ処理を補助するものとして、ルーティングエントリ管理やフィルタリングルール管理などのハードウェア処理ではカバーできない処理を肩代わりする。つまり、分類器１５がメイン機能を担当し、ＣＰＵはサブとして動作する。

【０２２２】

本発明の分類装置１５は、状態遷移によるテーブルルックアップ駆動のプロセッサにも適用可能である。図７１は単純状態遷移を示しており、ｋｅｙｉを検索し、その結果ＴＡＧ（ルール）でｋｅｙｉ＋１を読み取れるようにしておけば、ｋｅｙ１－ｋｅｙ２－・・・－ｋｅｙｊ－ｋｅｙ１のような単純ループ型の状態遷移ができる。

【０２２３】

図７２は、イベント連動型単純状態遷移を示しており、イベントｉがトリガとなってｋｅｙｉ＋１に遷移しイベントｉ以外でｋｅｙｉに戻る。ｋｅｙｉはイベントｉ待ち行列でイベントｉを待つ。イベントｉが来たらｋｅｙｉにイベントｉを直列につないで検索する。イベントｉは１ビットで、正が｛1｝、誤が｛0｝としｋｅｙｉを「１０１・・・０１」とすれば「１０１・・・０１１」でヒットし、「１０１…０１０」でミスヒットとなる。ヒットルールにはイベントｉ＋１待ち行列アドレス、ｋｅｙｉ＋１が書き込んである。ミスヒットルールにはイベントｉ待ち行列アドレス、ｋｅｙｉが書き込んであり、戻ることになる。さらに、本発明の分類装置は、条件分岐を含む状態遷移にも適用可能である。

【０２２４】

また、知的連想マシン、オートマトンへの応用（言語理論・オートマトン理論

）、大規模なデータベースから有用なルールあるいはパターンを高速に発見するデータマイニングと呼ばれる分野などでも有効であり、応用範囲は非常に広い。

【０２２５】

【発明の効果】

以上に説明したように、本発明に係る照合方法および装置は、完全一致検索、最長一致検索から多次元範囲一致検索までカバーでき、複数の照合装置を備えた分類装置は、パケットフィルタリング・ＳＰＤ（Ｓｅｃｕｒｉｔｙ　Ｐｏｌｉｃｙ　Ｄａｔａｂａｓｅ）検索、ステートフルインスペクションなどのルール検索に用いられることができる。さらに、データキャッシュ、履歴キャッシュを導入することにより、照合速度を向上することが可能であり、適用範囲は広い。検索以外の応用機能としては、集合抽出、集合演算、整列（ソート）・ユニーク化、メモリアドレッシング、状態遷移マシン、データマイニングなどを挙げることができる。

【図面の簡単な説明】

【図１】　本発明に係る管理装置の概要を示す図である。

【図２】　一致検索の照合状態を説明する図である。

【図３】　バイナリビットツリー上の照合状態を示す図である。

【図４】　右倒れ一致型の近似について説明する図である。

【図５】　左倒れ一致型の近似について説明する図である。

【図６】　バイナリビットツリーをテーブル分割した様子を示す図である。

【図７】　照合テーブル（部分テーブル）を示す図である。

【図８】　ＭＩＮ／ＭＡＸサーチを説明する図である。

【図９】　部分テーブルでＭＩＮ／ＭＡＸサーチを行う様子を示す図である。

【図１０】　圧縮テーブルでＭＩＮ／ＭＡＸサーチを行う様子を示す図である。

【図１１】　バイパス処理を示す図である。

【図１２】　照合過程でバックトラックが発生する様子を示す図である。

【図１３】　バックトラックが発生した後に、バイパス処理を実行する様子

を示す図である。

【図１４】　検索装置の概要を示す図である。

【図１５】　メモリ共用型の分類装置を示す図である。

【図１６】　メモリ分離型の分類装置を示す図である。

【図１７】　繰り返し型の分類装置を示す図である。

【図１８】　照合装置の概要を示す図である。

【図１９】　照合装置の処理を示すフローチャートである。

【図２０】　照合装置における処理の進行を説明する図である。

【図２１】　比較ユニットを示す図である。

【図２２】　比較器の論理を示す図である。

【図２３】　比較ユニットの異なる例を示す図である。

【図２４】　マスク器を示す図である。

【図２５】　マスク器の論理を示す図である。

【図２６】　選択回路の一致論理を示す図である。

【図２７】　選択回路の範囲検索（右倒れ）の論理を示す図である。

【図２８】　選択回路の範囲検索（左倒れ）の論理を示す図である。

【図２９】　比較ユニット、マスク器、選択回路の出力（右倒れ）を示す図である。

【図３０】　比較ユニット、マスク器、選択回路の出力（左倒れ）を示す図である。

【図３１】　コマンドの例を示す図である。

【図３２】　バイパス処理用のバスでバックトラックを処理する様子を示す図である。

【図３３】　メモリ共有型の分類装置でバックトラックを処理する様子を示す図である。

【図３４】　分類装置においてスルーを処理する様子を示す図である。

【図３５】　異なる方式でスルーを処理する様子を示す図である。

【図３６】　さらに異なる方式でスルーを処理する様子を示す図である。

【図３７】　さらに異なる方式でスルーを処理する様子を示す図である。

【図３８】　待ち合わせ方式でエントリを追加する処理を示す図である。

【図３９】　待ち合わせ方式でパイプラインに沿って処理を進める様子を示す図である。

【図４０】　待ち合わせ方式でエントリを追加する更新ユニットの機能を示す図である。

【図４１】　待機管理ユニットの機能を示す図である。

【図４２】　双方向リスト方式でエントリを追加する処理を示す図である。

【図４３】　双方向リスト方式でパイプラインに沿って処理を進める様子を示す図である。

【図４４】　双方向リスト方式でエントリを追加する更新ユニットの機能を示す図である。

【図４５】　待ち合わせ方式でエントリを削除する処理を示す図である。

【図４６】　待ち合わせ方式でエントリを削除する更新ユニットの機能を示す図である（コマンドがＤＥＬのとき）。

【図４７】　待ち合わせ方式でエントリを削除する更新ユニットの機能を示す図である（コマンドがＤＥＬＵＰＤＴのとき）。

【図４８】　待ち合わせ方式でコマンドがＤＥＬＲＤＢＰおよびＤＥＬＷＲＢＰの処理を示す図である。

【図４９】　双方向リスト方式でエントリを削除する更新ユニットの機能を示す図である（コマンドがＤＥＬのとき）。

【図５０】　双方向リスト方式でエントリを削除する更新ユニットの機能を示す図である（コマンドがＤＥＬＵＰＤＴのとき）。

【図５１】　双方向リスト方式でコマンドがＤＥＬＲＤＢＰおよびＤＥＬＷＲＢＰの処理を示す図である。

【図５２】　データキャッシュを備えた分類装置を示す図である。

【図５３】　データキャッシュのラインサイズを示す図である。

【図５４】　データキャッシュの管理を示す図である。

【図５５】　メモリの管理を示す図である。

【図５６】　空テーブル管理キューの応答を示す図である。

【図５７】　ＳＤＲＡＭ共用型履歴キャッシュシステムを備えた分類装置を示す図である。

【図５８】　ＳＤＲＡＭ共用型履歴キャッシュシステムの概要を示す図である。

【図５９】　履歴ＳＲＡＭキャッシュ付き分類装置を示す図である。

【図６０】　履歴ＳＲＡＭキャッシュの概要を示す図である。

【図６１】　ＡＮＤ化器における行列演算を説明する図である。

【図６２】　行列演算の一例を示す図である。

【図６３】　行列演算の左行列を示す図である。

【図６４】　ＬＳＩ化に適して行列演算の概要を示す図である。

【図６５】　ＡＮＤ化器のハードウェアを示す図である。

【図６６】　下位テーブルを省略する様子を示した図である。

【図６７】　要求応答方式のメモリアクセスを採用した照合装置の概要を示す図である。

【図６８】　要求応答方式のメモリアクセスを採用した照合装置の異なる例を示す図である。

【図６９】　照合テーブルの改良型を示す図である。

【図７０】　非リニアメモリアドレッシングを示す図である。

【図７１】　単純状態遷移を示す図である。

【図７２】　イベント連動型単純状態遷移を示す図である。

【符号の説明】

１　　データ管理装置

１０　　検索装置

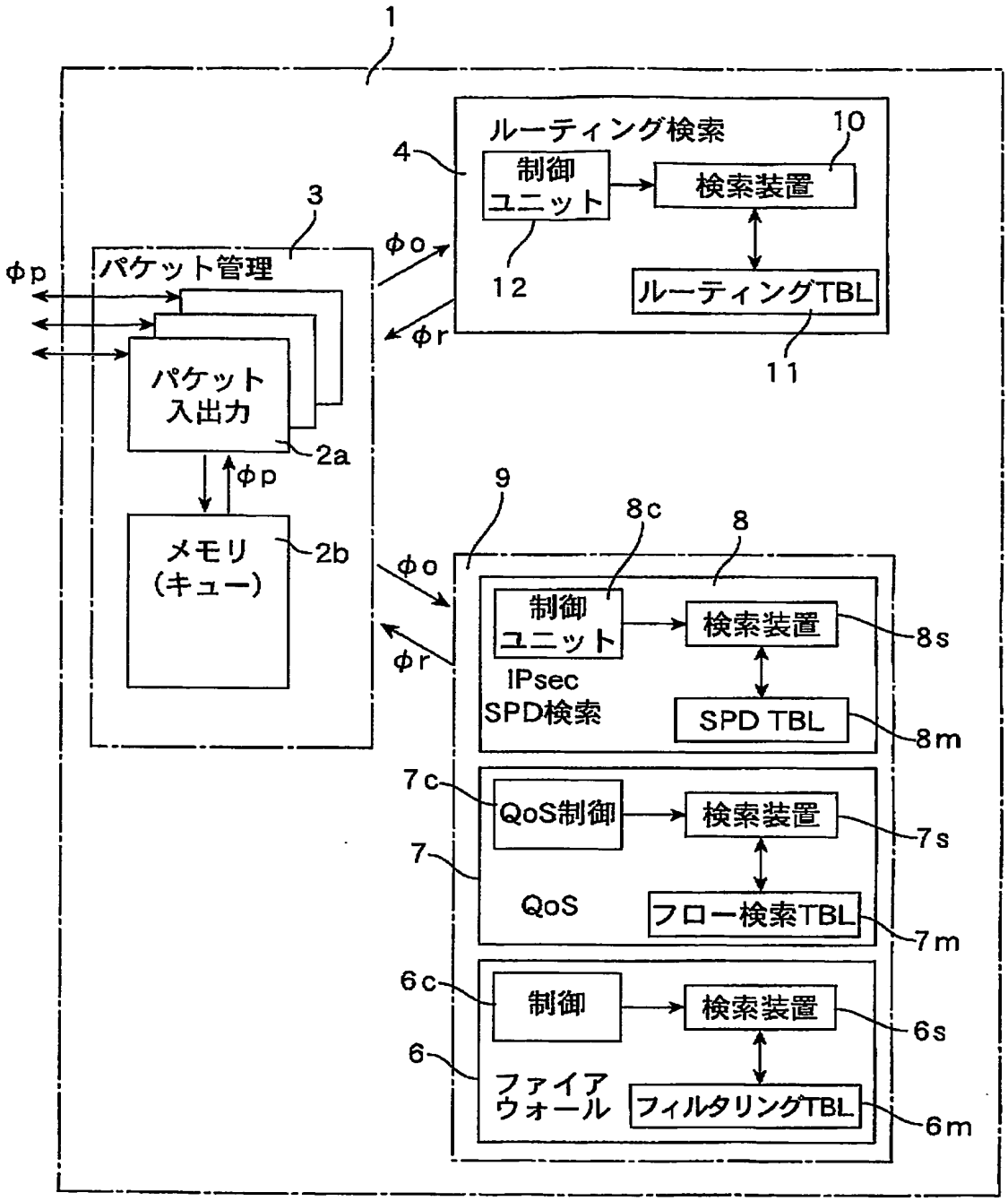
１１　　ＳＤＲＡＭ（メモリ）

１５　　分類装置

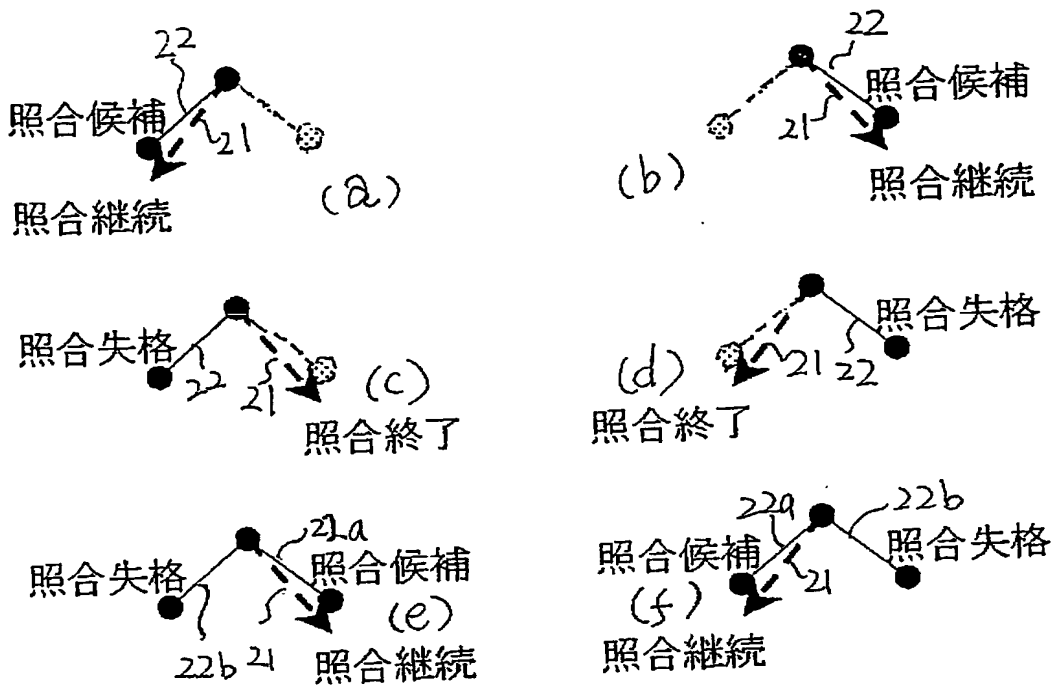
５０　　照合装置

【書類名】　　図面

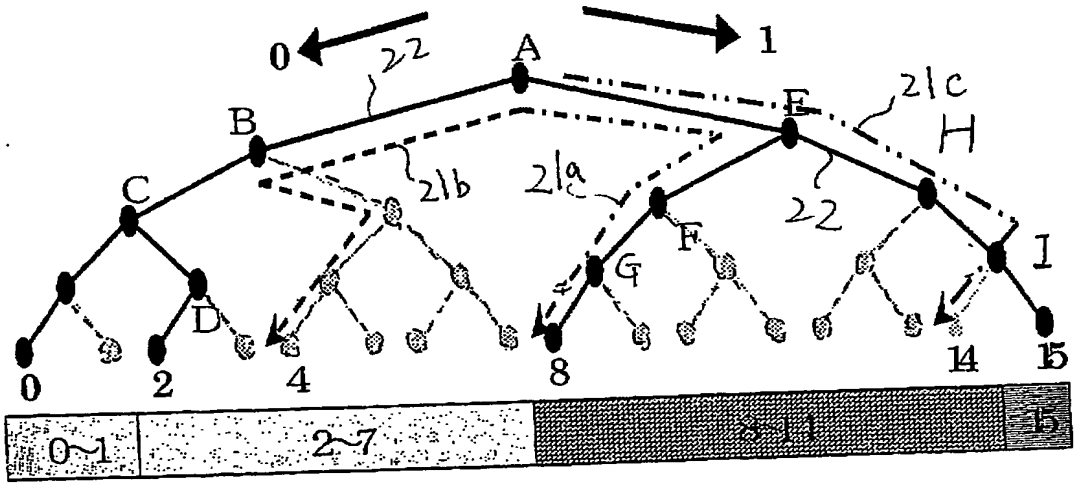
【図１】


1
ルーティング検索
10
4
制御
ユニット
検索装置
3
φo
φp
パケット管理
12
ルーティングTBL
φr
11
パケット
入出力
2a
φp
9
8c
8
メモリ
(キュー)
2b
φo
制御
ユニット
検索装置
8s
φr
IPsec
SPD検索
SPD TBL
8m
7c
QoS制御
検索装置
7s
7
QoS
フロー検索TBL
7m
6c
制御
検索装置
6s
6
ファイア
ウォール
フィルタリングTBL
6m

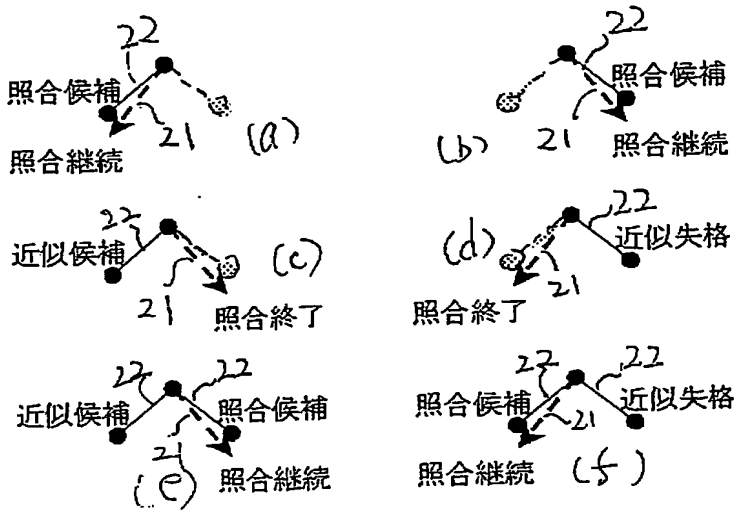
【図２】


22
照合候補
21
照合継続
(a)
22
照合候補
21
(b)
照合継続
照合失格
22
21
(c)
照合終了
照合失格
(d)
21 22
照合終了
照合失格
21a
照合候補
(e)
22b 21
照合継続
22a
22b
照合失格
照合候補
21
(f)
照合継続

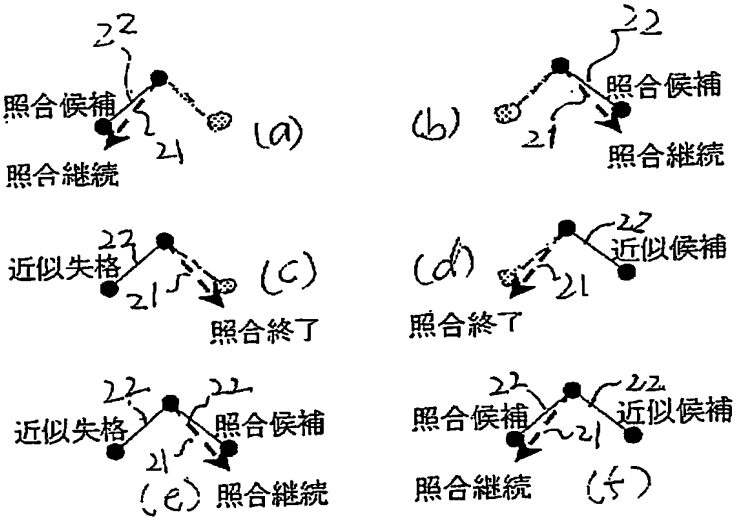
【図３】


0
22
A
1
B
E
21c
H
C
21b
21a
22
F
G
I
D
0
2
4
8
14
15
0~1
2~7
8~14
15

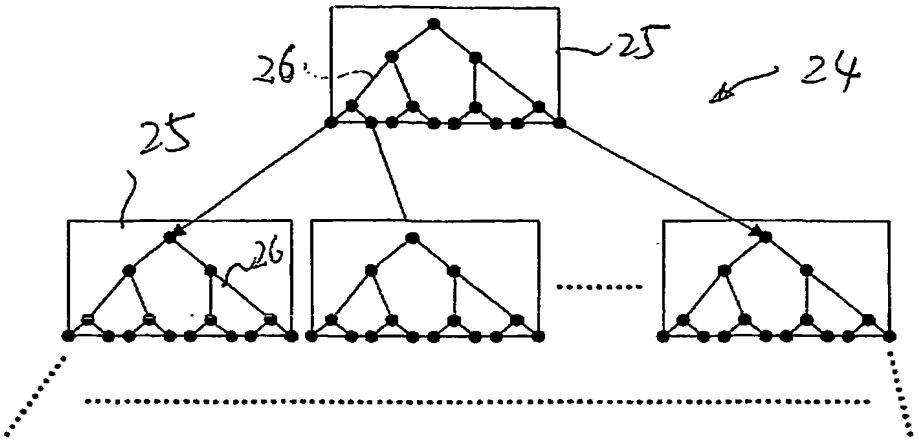
【図４】


22
照合候補
21
(a)
照合継続
22
照合候補
(b)
21
照合継続
22
近似候補
(c)
21
照合終了
22
(d)
近似失格
21
照合終了
22
22
近似候補
照合候補
21
(e)
照合継続
22
22
照合候補
近似失格
21
照合継続
(f)

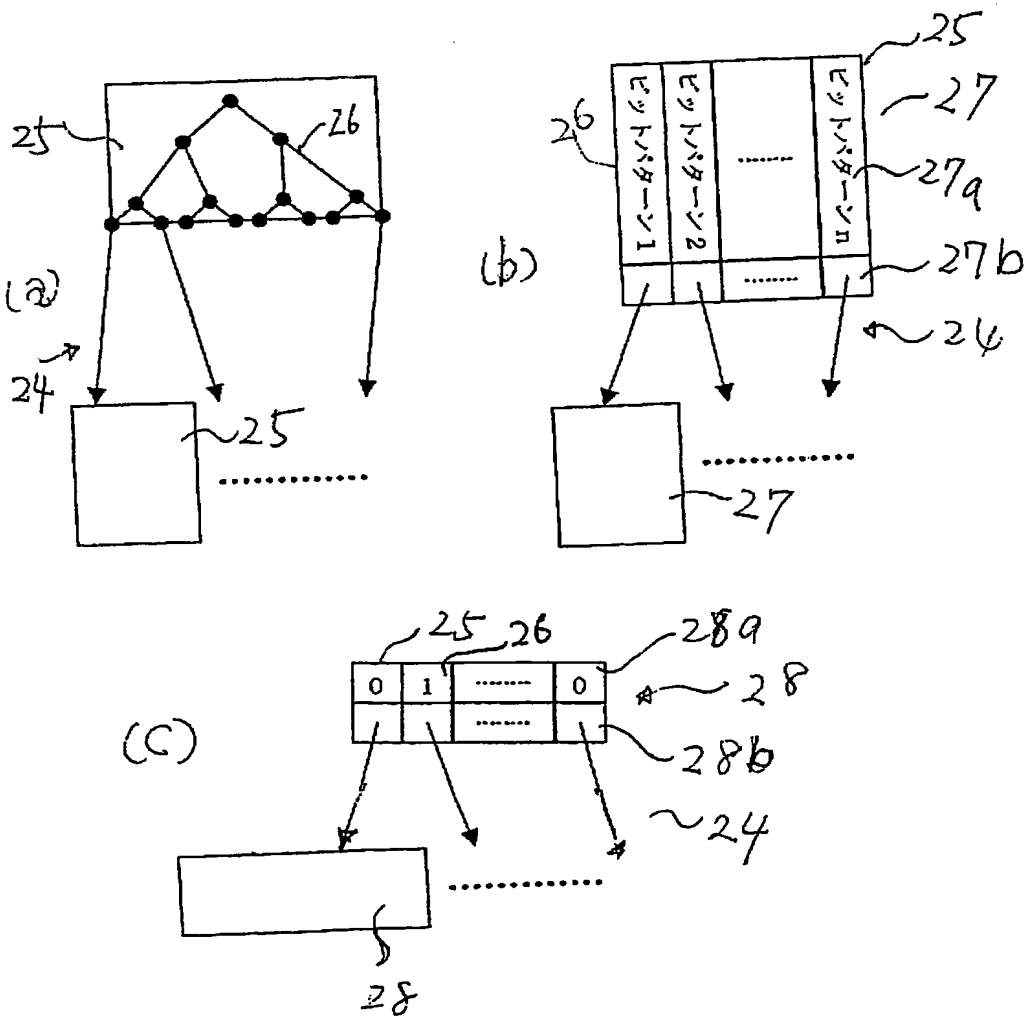
【図５】


22
照合候補
21
(a)
照合継続
22
(b)
照合候補
21
照合継続
22
近似失格
21
(c)
照合終了
22
(d)
近似候補
21
照合終了
22
22
近似失格
照合候補
21
(e)
照合継続
22
22
照合候補
近似候補
21
照合継続
(f)

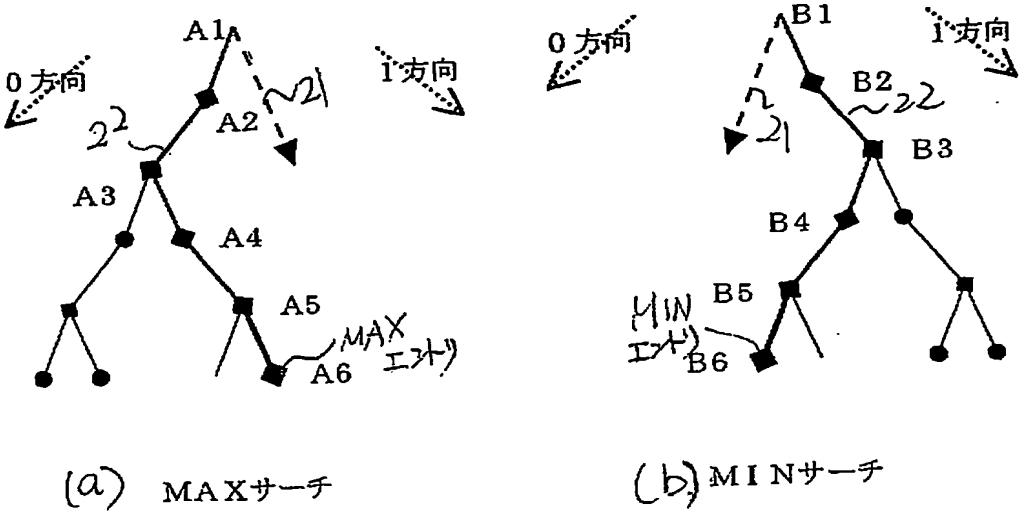
【図６】


26
25
24
25
26

【図７】

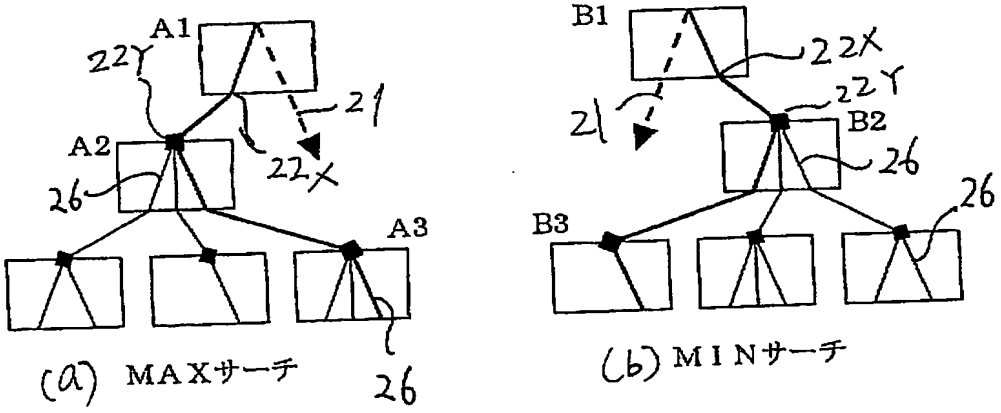

25
26
(a)
24
25
(b)
25
ビットパターン1
ビットパターン2
ビットパターンn
27
26
27a
27b
24
27
(c)
25
26
28a
0
1
0
28
28b
24
28

【図８】



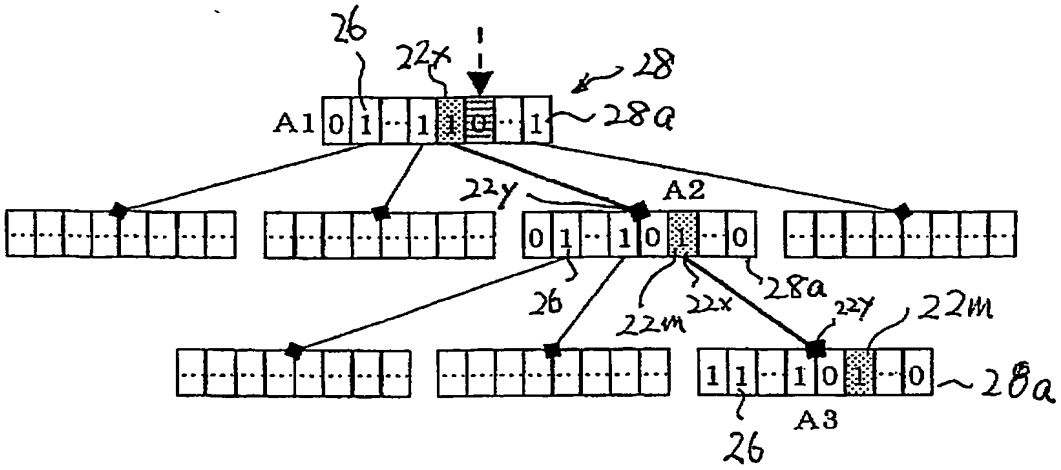
(a) MAXサーチ　　(b) MINサーチ

【図９】

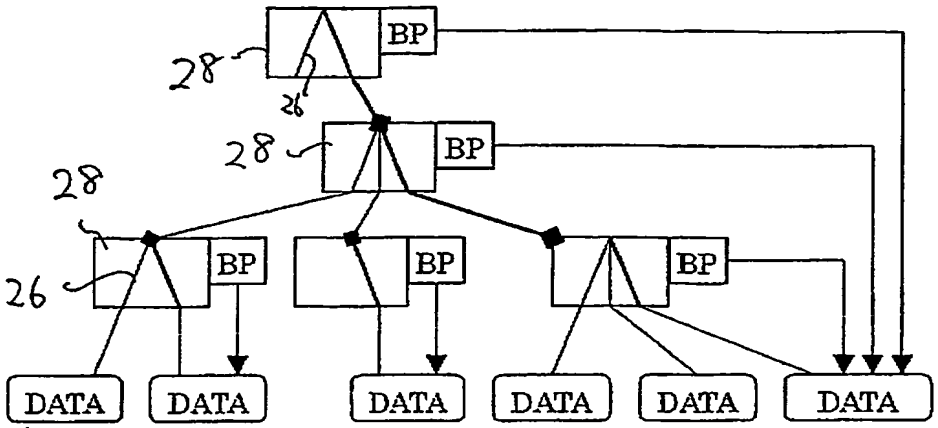


(a) MAXサーチ　　(b) MINサーチ

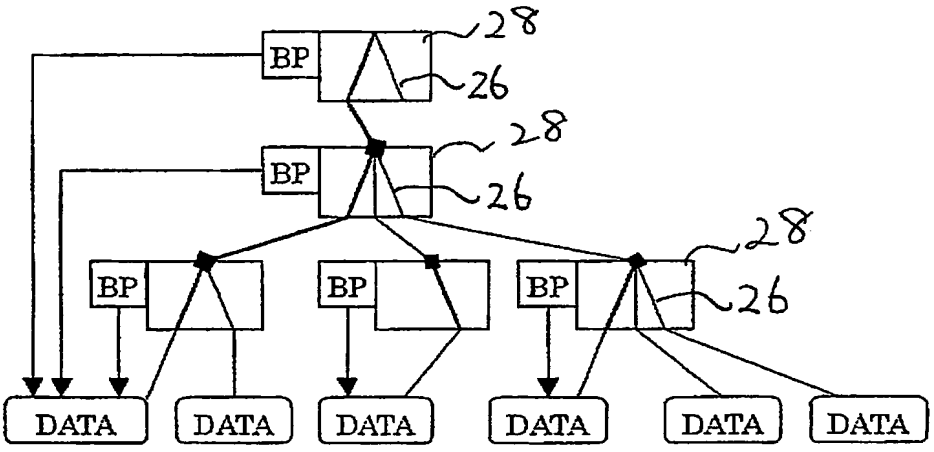
【図１０】


26
22x
28
A1
0 1 1 1 0 1
28a
22y
A2
0 1 1 0 0
26
22m
22x
28a
22y
22m
1 1 1 0 0
28a
A3
26

【図１１】

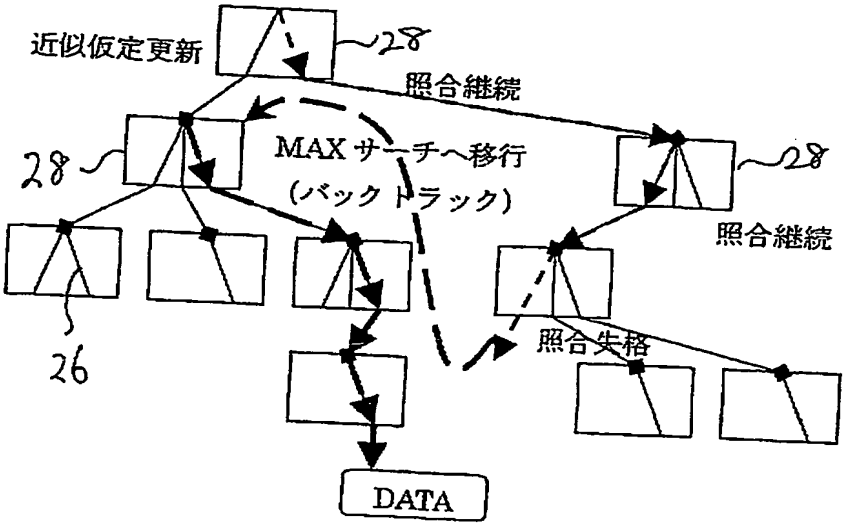


(a)　MAXバイパス

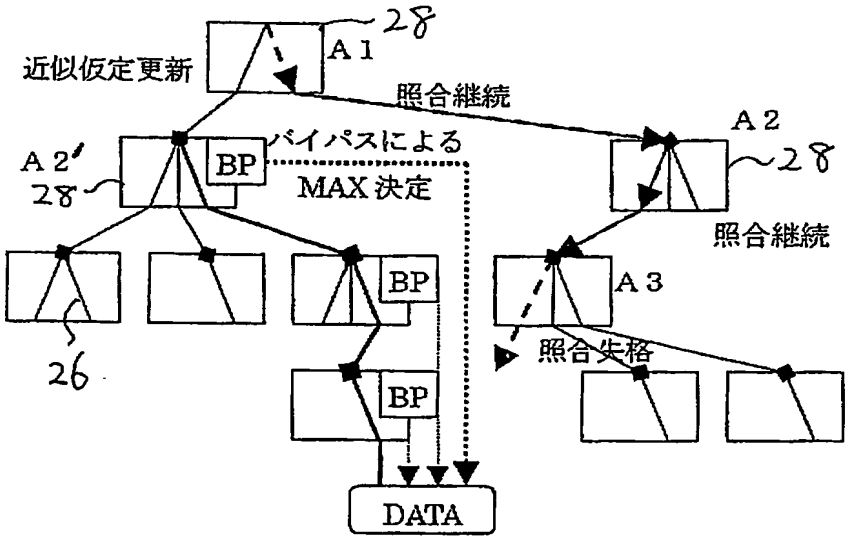


(b)　MINバイパス

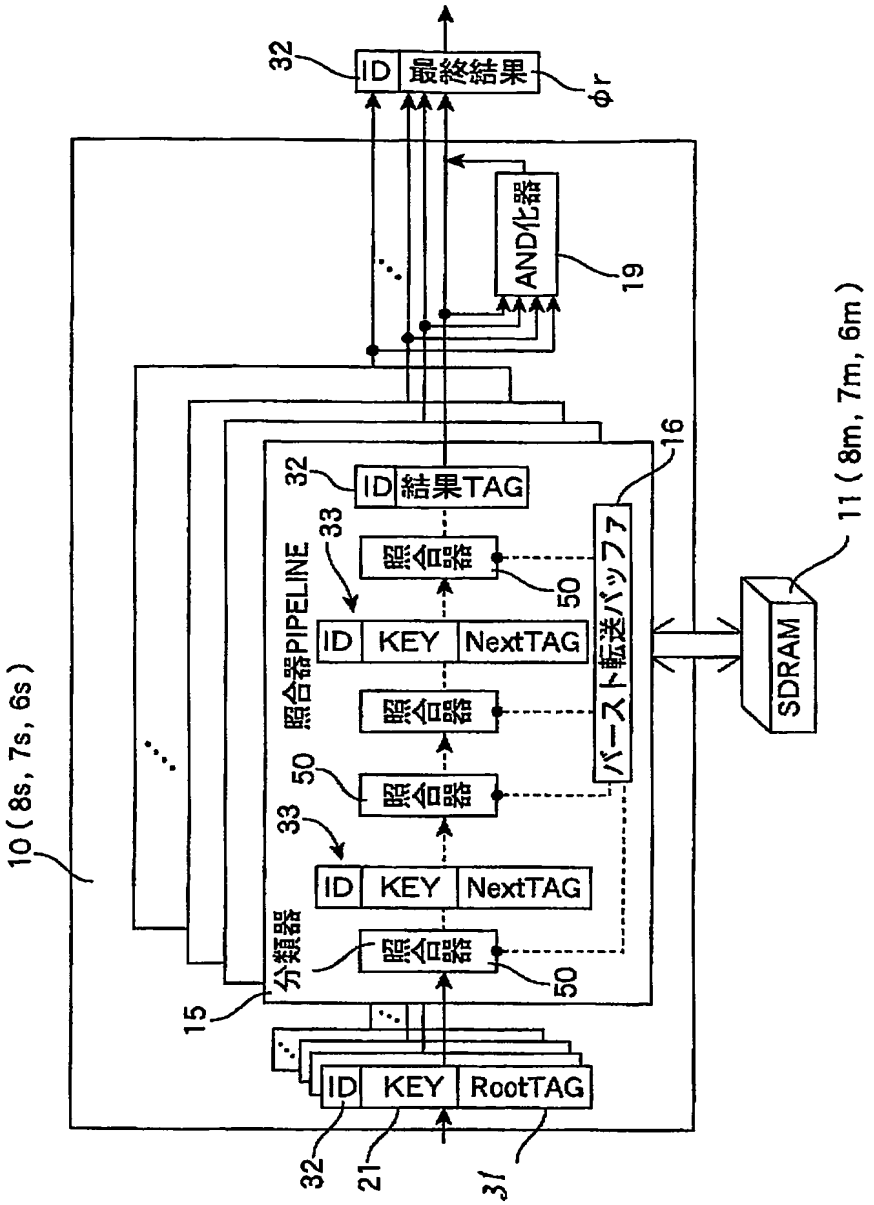
【図１２】


近似仮定更新
28
照合継続
28
MAX サーチへ移行
(バックトラック)
28
照合継続
26
照合失格
DATA

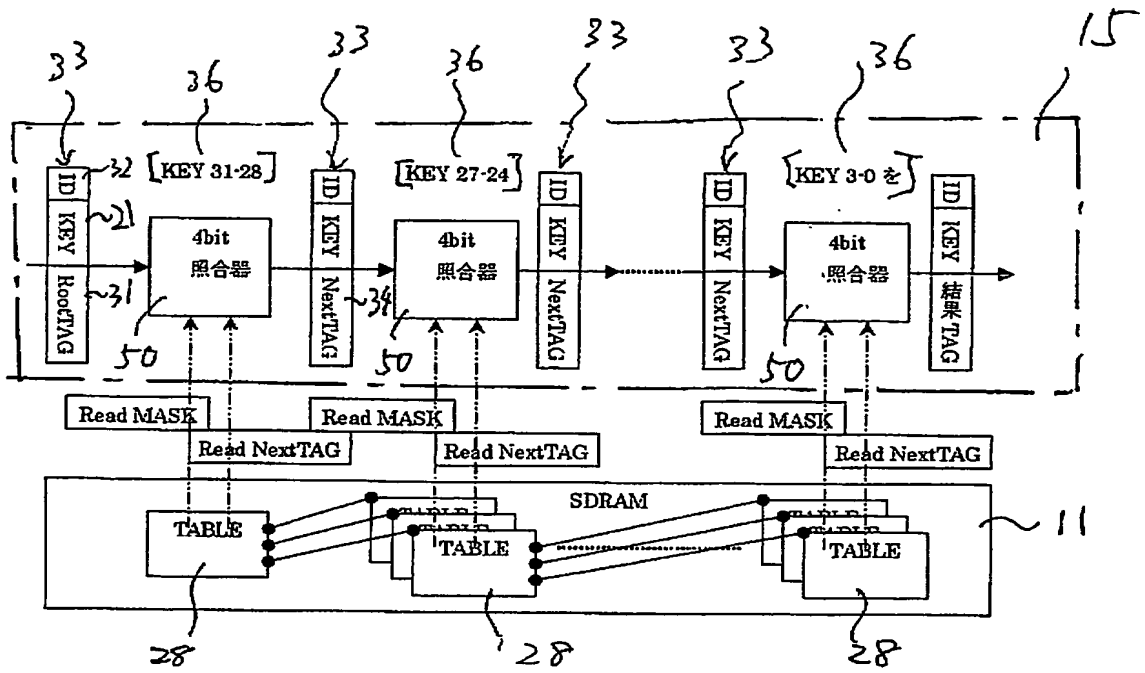
【図１３】


28
A１
近似仮定更新
照合継続
A２
A２'
28
BP
バイパスによる
MAX 決定
28
照合継続
BP
A３
26
照合失格
BP
DATA

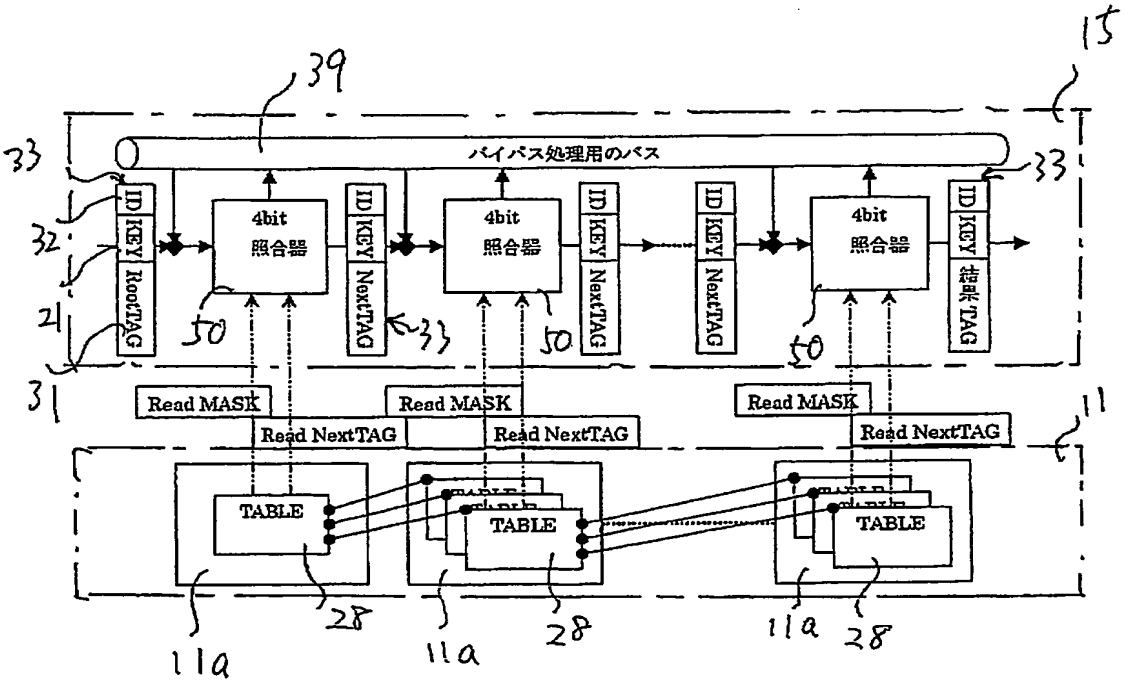
【図１４】


32
ID
最終結果
φr
AND化器
19
16
32
ID
結果TAG
33
照合器
50
照合器PIPELINE
ID
KEY
NextTAG
照合器
50
バースト転送バッファ
11 (8m, 7m, 6m)
SDRAM
10 (8s, 7s, 6s)
33
ID
KEY
NextTAG
分類器
照合器
50
15
ID
KEY
RootTAG
32
21
31

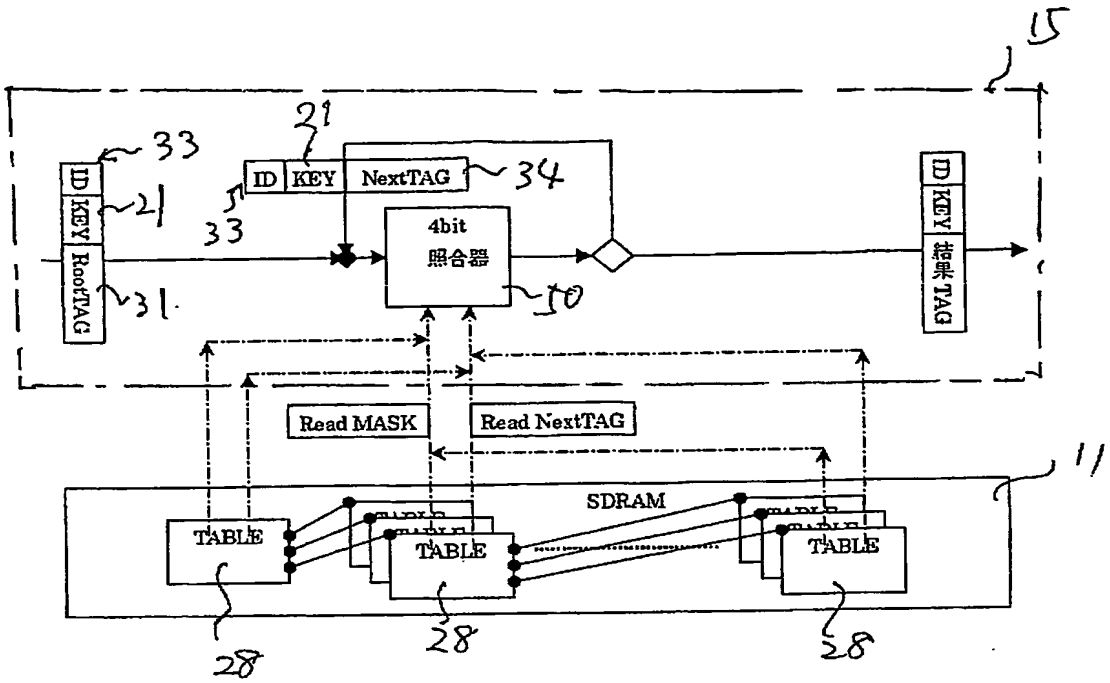
【図１５】


33
36
KEY 31-28
KEY 27-24
KEY 3-0 を
ID
KEY
RootTAG
NextTAG
結果TAG
4bit
照合器
32
21
31
34
50
15
Read MASK
Read NextTAG
SDRAM
TABLE
11
28

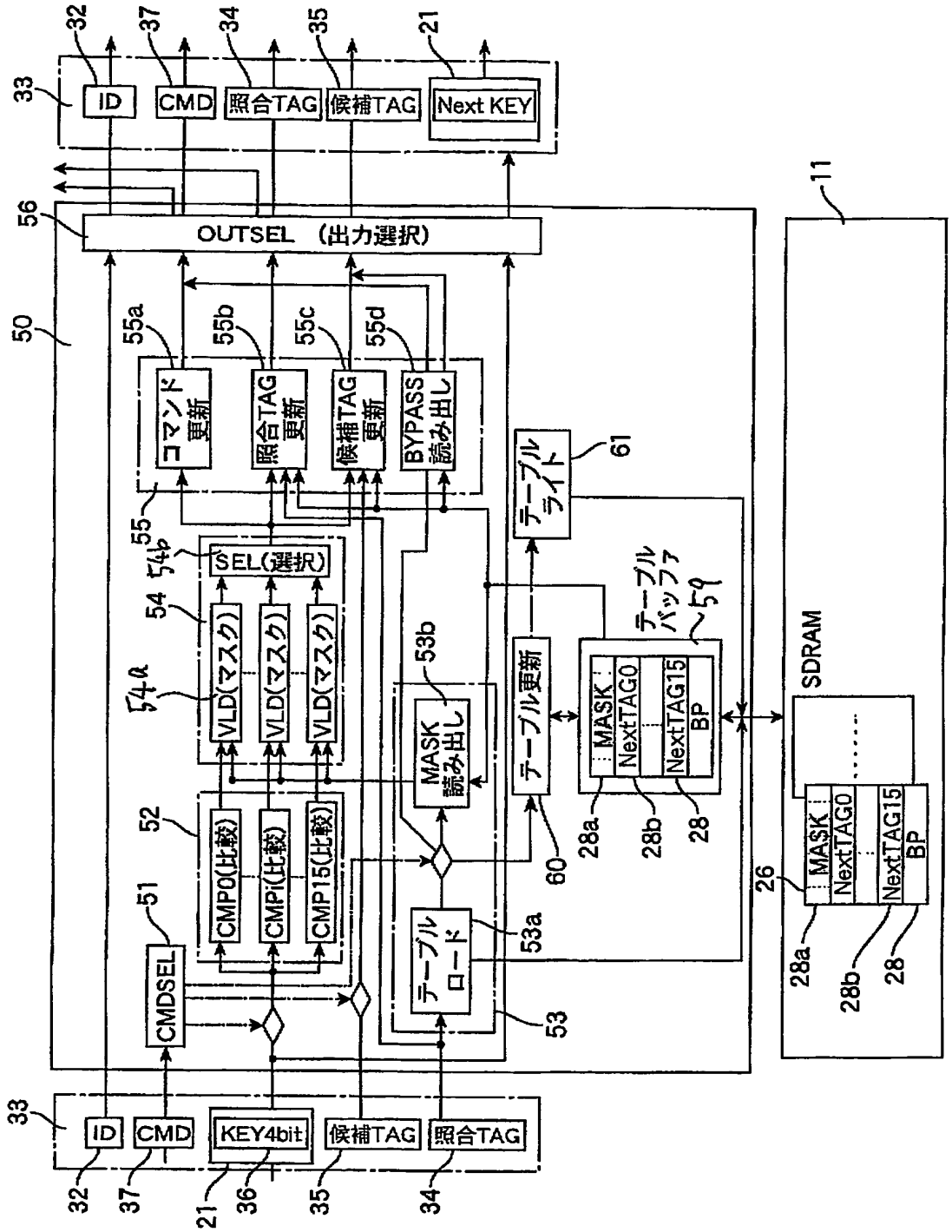
【図１６】


15
39
33
バイパス処理用のバス
ID
KEY
RootTAG
NextTAG
結果TAG
4bit
照合器
32
21
31
50
Read MASK
Read NextTAG
TABLE
11
11a
28

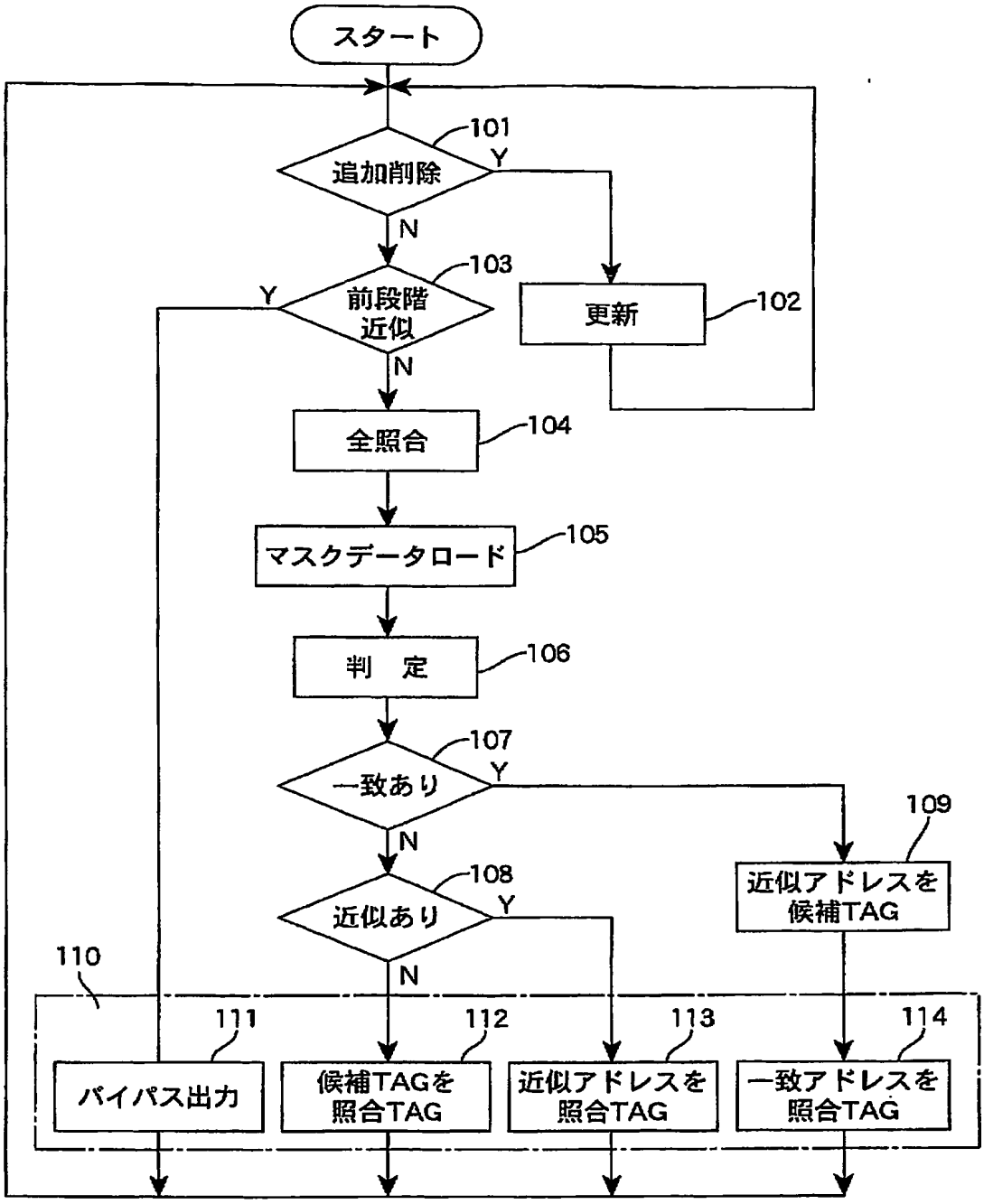
【図１７】


15
21
33
ID
KEY
NextTAG
34
33
ID
KEY
RootTAG
21
31
4bit
照合器
50
ID
KEY
結果TAG
Read MASK
Read NextTAG
11
SDRAM
TABLE
TABLE
TABLE
28
28
28

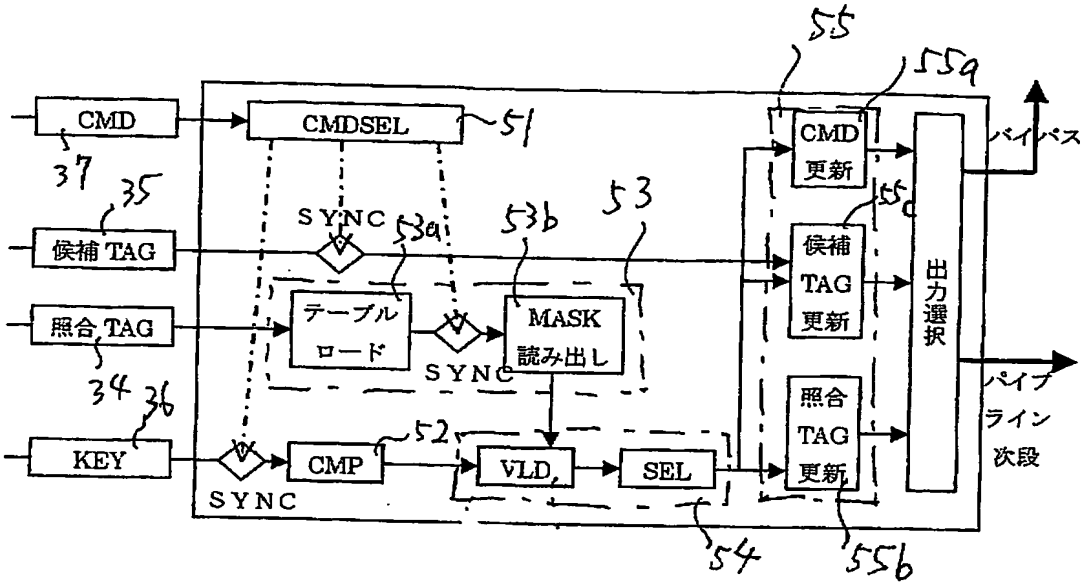
【図１８】


32
37
34
35
21
33
ID
CMD
照合TAG
候補TAG
Next KEY
56
OUTSEL　(出力選択)
11
50
55a
55b
55c
55d
コマンド更新
照合TAG更新
候補TAG更新
BYPASS読み出し
テーブルライト
61
55
54b
SEL(選択)
54
54a
VLD(マスク)
VLD(マスク)
VLD(マスク)
53b
MASK読み出し
テーブル更新
テーブルバッファ
59
MASK
NextTAG0
NextTAG15
BP
SDRAM
52
CMP0(比較)
CMPi(比較)
CMP15(比較)
51
CMDSEL
テーブルロード
53a
53
60
28a
28b
28
26
33
ID
CMD
KEY4bit
候補TAG
照合TAG
32
37
21
36
35
34

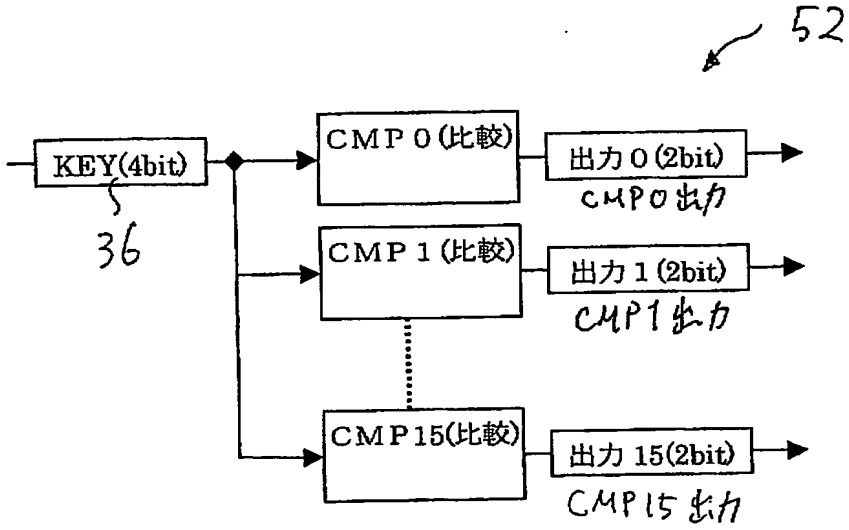
【図１９】


スタート
101
追加削除
Y
N
103
前段階
近似
Y
N
更新
102
全照合
104
マスクデータロード
105
判　定
106
107
一致あり
Y
N
109
近似アドレスを
候補TAG
108
近似あり
Y
N
110
111
112
113
114
バイパス出力
候補TAGを
照合TAG
近似アドレスを
照合TAG
一致アドレスを
照合TAG

【図２０】


CMD
37
CMDSEL
51
55
55a
CMD
更新
バイパス
35
候補 TAG
SYNC
53a
53b
53
候補
TAG
更新
55c
出力選択
照合 TAG
34
テーブル
ロード
SYNC
MASK
読み出し
照合
TAG
更新
パイプ
ライン
次段
36
KEY
52
CMP
VLD
SEL
SYNC
54
55b

【図２１】

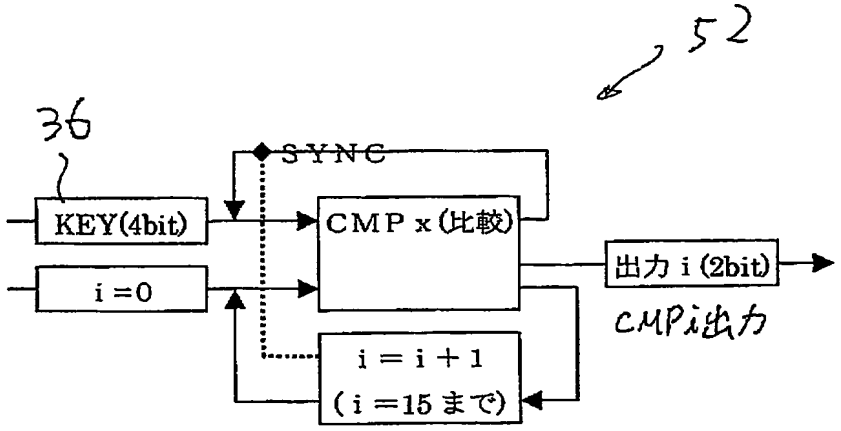

52
KEY(4bit)
36
CMP０(比較)
出力０(2bit)
CMP0出力
CMP１(比較)
出力１(2bit)
CMP1出力
CMP15(比較)
出力 15(2bit)
CMP15出力

【図２２】

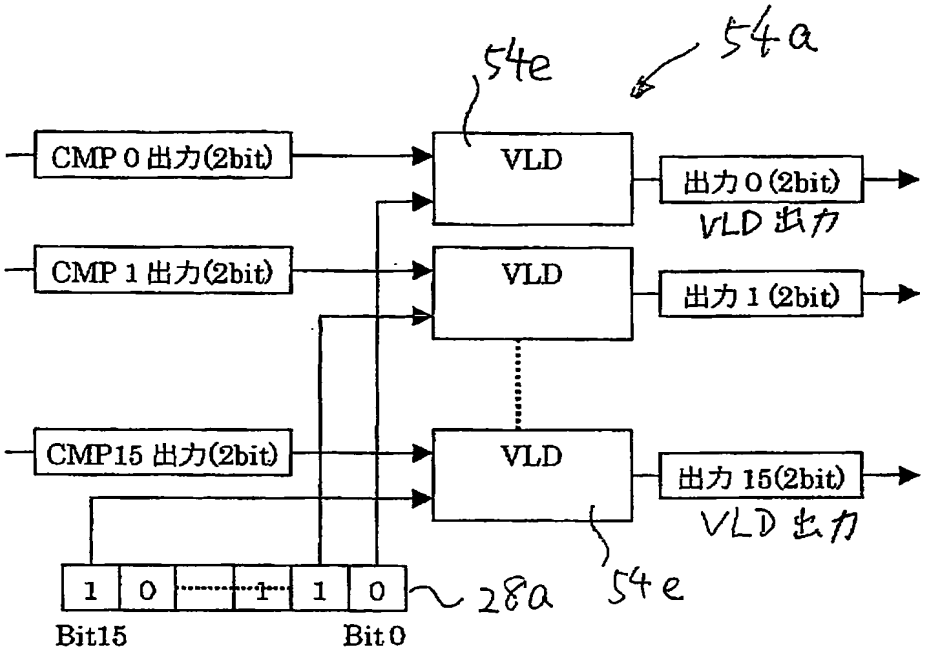
比較器ＣＭＰｉの演算論理

ＣＭＰｉ　入力ｘ	出力
ｘ＜ｉ	10（「＜」）
ｘ＝ｉ	01（「＝」）
ｘ＞ｉ	11（「＞」）
ｉ＝０～15（固定値） ｘ＝０～15（４bit入力値）	

【図２３】



【図２４】



【図２５】

マスク器ＶＬＤの演算論理

CMP i 出力	MASK値	ＶＬＤ出力
10	0	00 (「×」)
01	0	00 (「×」)
11	0	00 (「×」)
10	1	10 (「<」)
01	1	01 (「=」)
11	1	11 (「>」)
MASK 値による AND 演算		

【図２６】

完全一致検索の選択論理

ＶＬＤ出力	出力G_f
00（「×」）	0
10（「<」）	0
01（「=」）	1
11（「>」）	0

G_f（i）$=\overline{\mathrm{Bit1}}*\mathrm{Bit0}$

$S_f=\Sigma G_f$（i）

$N_f=G_f^{-1}$（1）

（G_f（i）＝1となるiを求める逆関数）

【図２７】

範囲検索の照合継続選択論理（右倒れ一致型）

ＶＬＤ出力	中間F_r	出力G_r
00（「×」）	0	0
10（「<」）	0	0
01（「=」）	0	0
11（「>」）	1	G_r（i）

F_r（i）＝Ｂｉｔ１＊Ｂｉｔ０

H_r（－１）＝０（漸化式仮想初期値）

H_r（i）＝H_r（i－１）＋F_r（i）

G_r（i）＝$\overline{H_r(i-1)}$ ＊H_r（i）

$S_r=\Sigma G_r$（i）

$N_r=G_r^{-1}$（１）

（G_r（i）＝１となるｉを求める逆関数）

H_r（i－１）	F_r（i）	H_r（i）
0	0	0
0	1	1
1	0	1
1	1	1

H_r（i－１）	H_r（i）	G_r（i）
0	0	0
0	1	1
1	0	0
1	1	0

【図２８】

範囲検索の選択論理（左倒れ一致）

ＶＬＤ出力	中間F_l	出力G_l
00（「×」）	0	0
10（「＜」）	1	G_l（i）
01（「＝」）	0	0
11（「＞」）	0	0

F_l（i）＝Ｂｉｔ１＊$\overline{\text{Ｂｉｔ０}}$

H_l（16）＝０（漸化式仮想初期値）

H_l（i）＝H_l（i＋1）＋F_l（i）

G_l（i）＝$\overline{H_l(i+1)}$＊H_l（i）

$S_l=\Sigma G_l$（i）

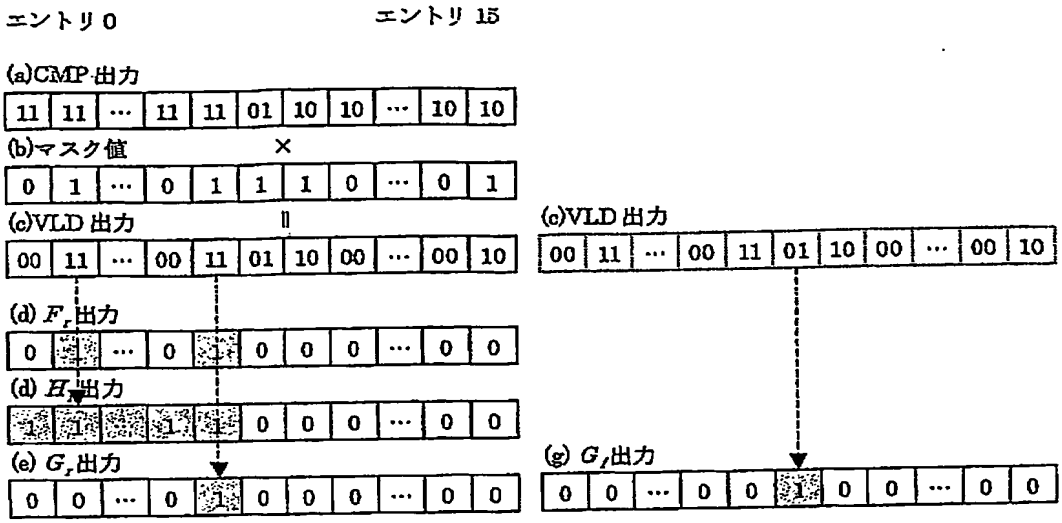
$N_l=G_l^{-1}$（1）

（G_l（i）＝1となるｉを求める逆関数）

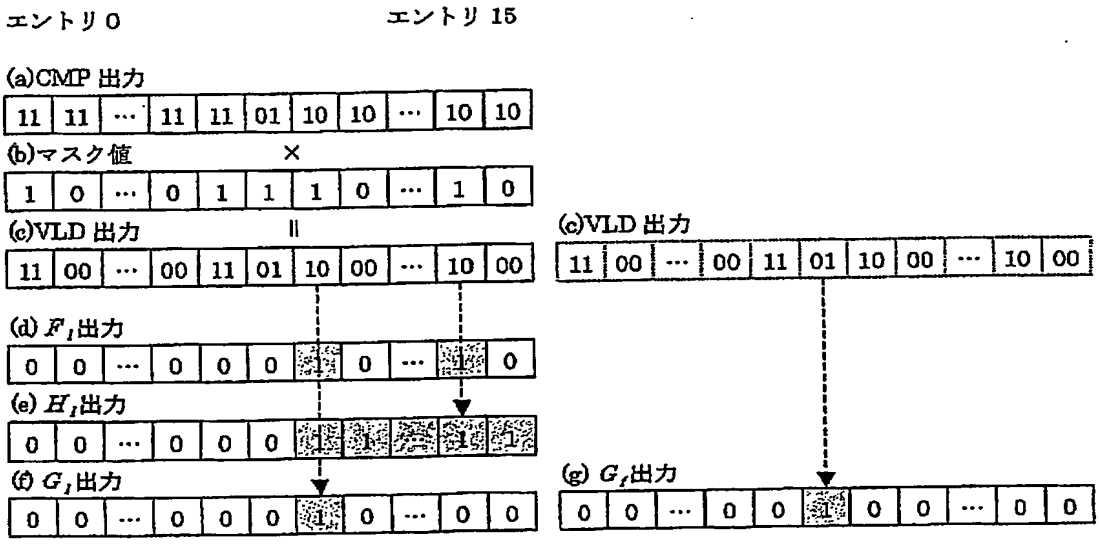
H_l（i＋1）	F_l（i）	H_l（i）
0	0	0
0	1	1
1	0	1
1	1	1

H_l（i＋1）	H_l（i）	G_l（i）
0	0	0
0	1	1
1	0	0
1	1	0

【図２９】


エントリ０
エントリ 15
(a)CMP 出力
11 11 … 11 11 01 10 10 … 10 10
(b)マスク値
×
0 1 … 0 1 1 1 0 … 0 1
(c)VLD 出力
=
00 11 … 00 11 01 10 00 … 00 10
(d) F_t 出力
0 1 … 0 1 0 0 0 … 0 0
(d) H_t 出力
1 1 … 1 1 0 0 0 … 0 0
(e) G_t 出力
0 0 … 0 1 0 0 0 … 0 0
(c)VLD 出力
00 11 … 00 11 01 10 00 … 00 10
(g) G_t出力
0 0 … 0 0 1 0 0 … 0 0

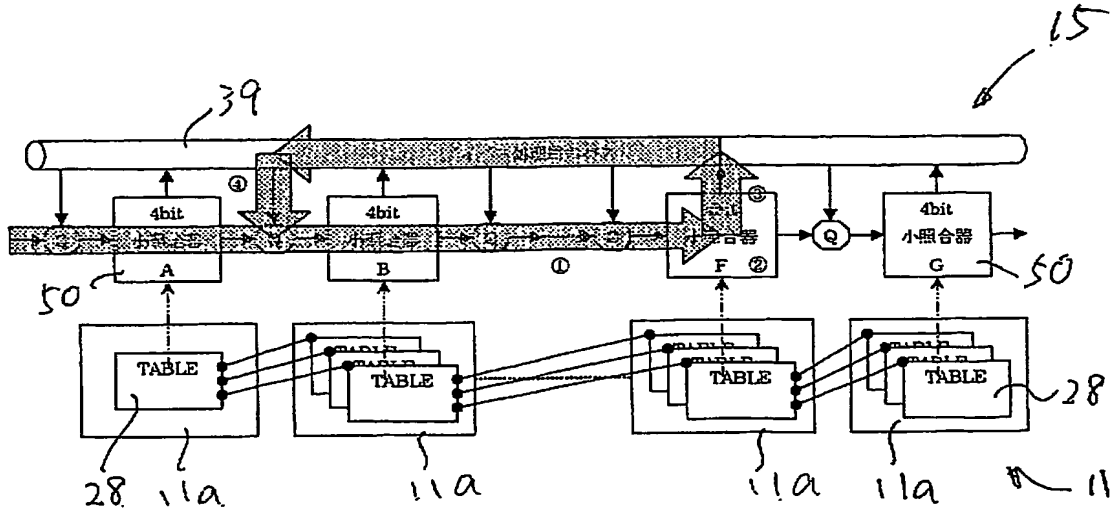
【図３０】


エントリ０
エントリ 15
(a)CMP 出力
11 11 … 11 11 01 10 10 … 10 10
(b)マスク値
×
1 0 … 0 1 1 1 0 … 1 0
(c)VLD 出力
=
11 00 … 00 11 01 10 00 … 10 00
(d) F_t出力
0 0 … 0 0 0 1 0 … 1 0
(e) H_t出力
0 0 … 0 0 0 1 1 … 1 1
(f) G_t出力
0 0 … 0 0 0 1 0 … 0 0
(c)VLD 出力
11 00 … 00 11 01 10 00 … 10 00
(g) G_t出力
0 0 … 0 0 1 0 0 … 0 0

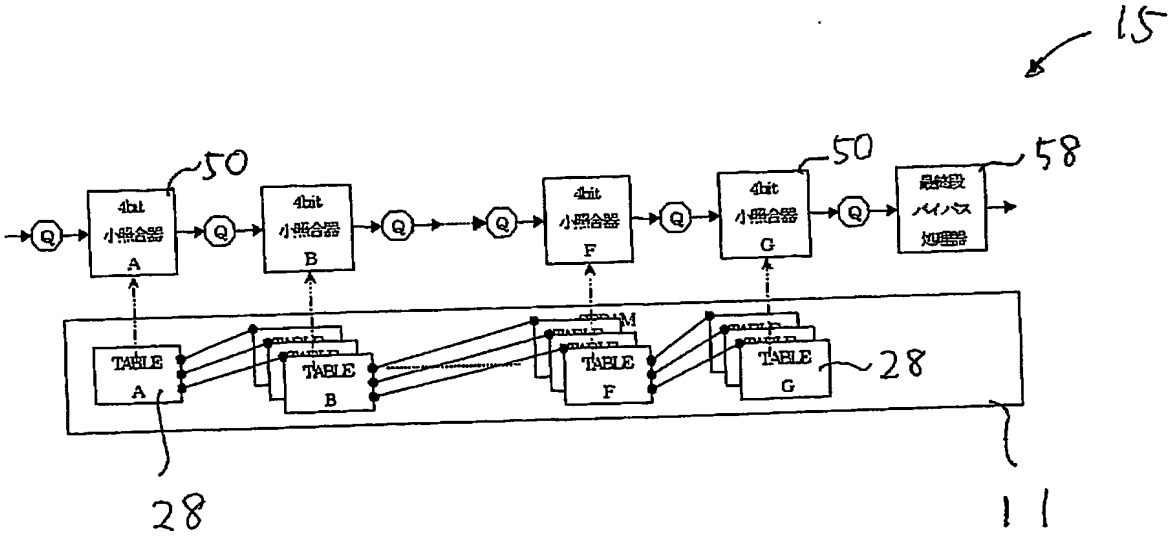
【図３１】

種別	コマンド	説明
完全一致検索	FMATCHEXEC	照合継続
	FMATCHHIT	ヒット
	FMATCHFAIL	ミスヒット
範囲検索	MATCHEXEC	照合継続
	MATCHRDBP	バイパス読み出し
	MATCHFHIT	完全一致ヒット
	MATCHNHIT	近似ヒット
	MATCHFAIL	ミスヒット
登録	ADD	上位→下位
	ADDNEW	新規テーブル
	ADDFIN	登録完了
	ADDFAIL	登録失敗
	ADDUPDT	BP 更新フィード
	(ADDWCLR)	(待ち取り消し)
削除	DEL	下位継続
	DELFIN	削除完了
	DELFAIL	削除失敗
	DELUPDT	BP 更新フィード
	DELRDBP	下位 BP 参照
	DELWRBP	自 BP 更新

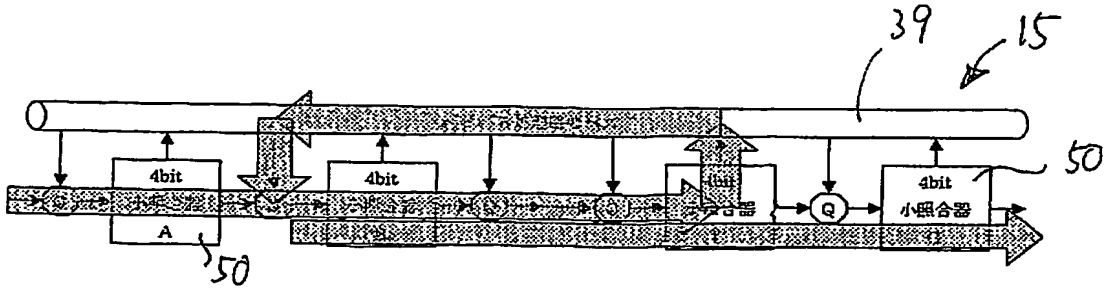
【図３２】


15
39
4bit
小照合器
A
B
F
G
50
TABLE
28
11a
11

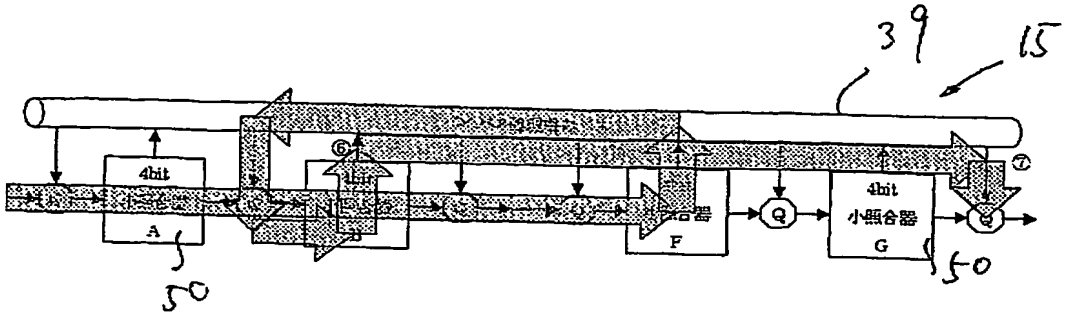
【図３３】


15
50
58
4bit
小照合器
A
B
F
G
最終段
バイパス
処理器
TABLE
28
11

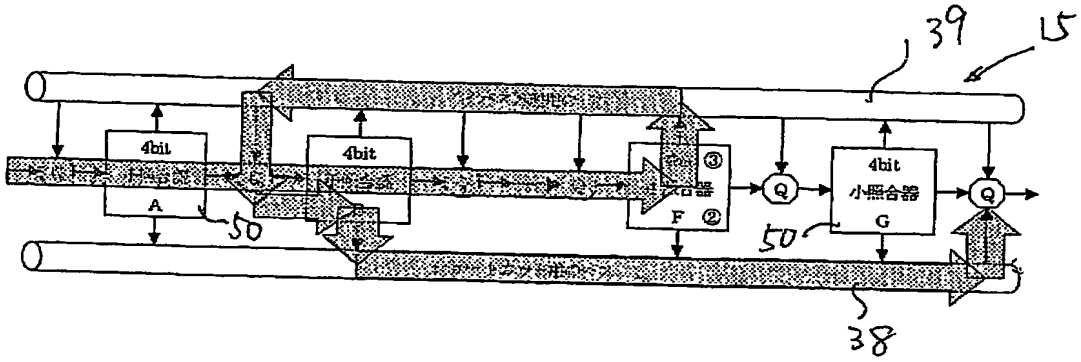
【図３４】


39
15
50
4bit
小照合器
A

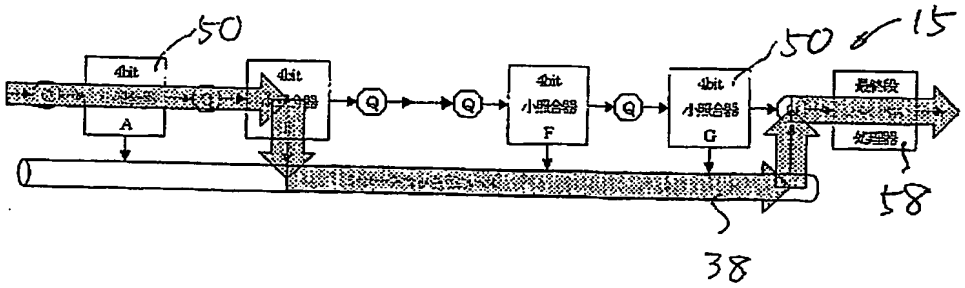
【図３５】


39
15
4bit
A
50
4bit
F
Q
4bit
小照合器
G
50

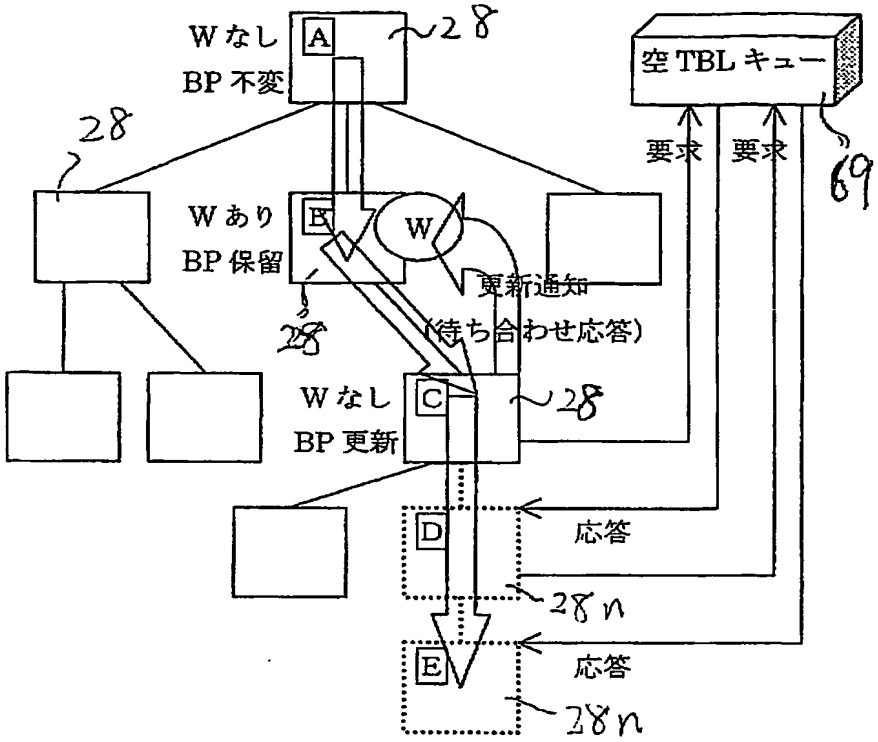
【図３６】


39
15
4bit
A
50
4bit
F
Q
4bit
小照合器
G
50
38

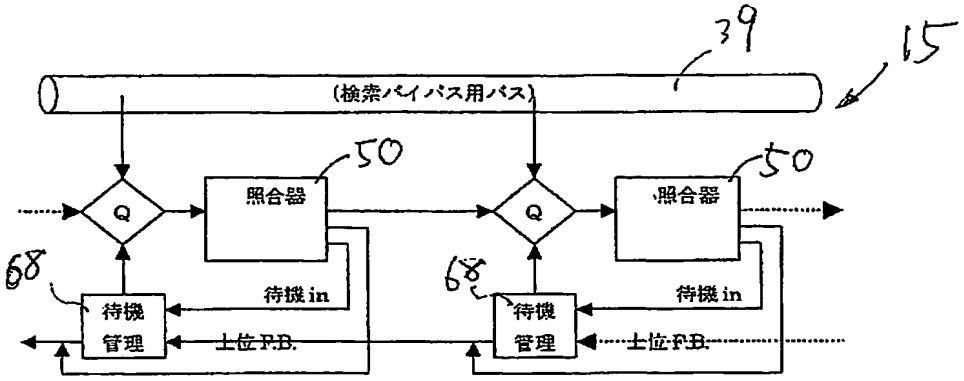
【図３７】


50
15
4bit
A
4bit
Q
4bit
小照合器
F
50
4bit
小照合器
G
58
38

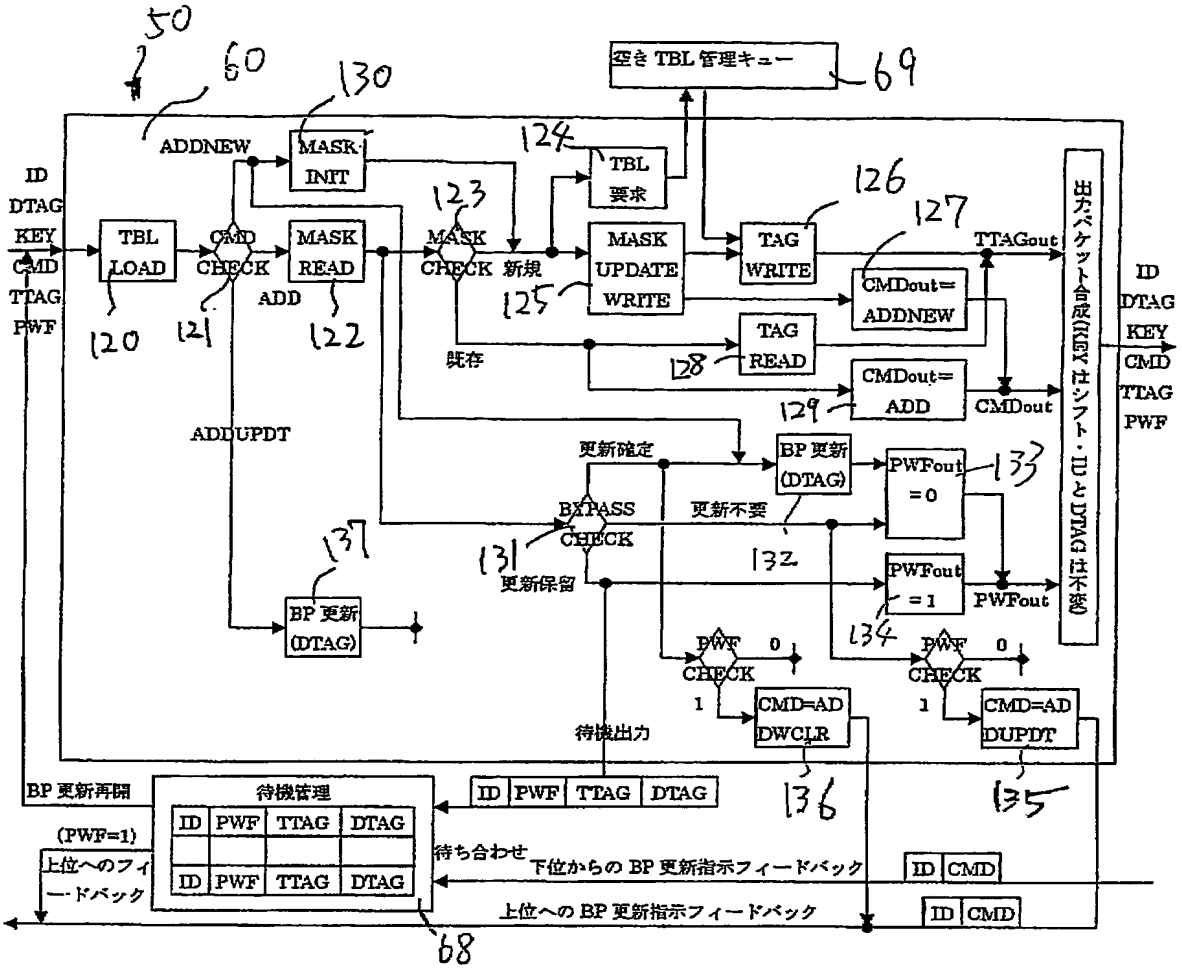
【図３８】


Wなし
BP不変
A
28
空TBLキュー
要求
要求
69
28
Wあり
BP保留
B
W
更新通知
(待ち合わせ応答)
28
Wなし
BP更新
C
28
D
応答
28n
E
応答
28n

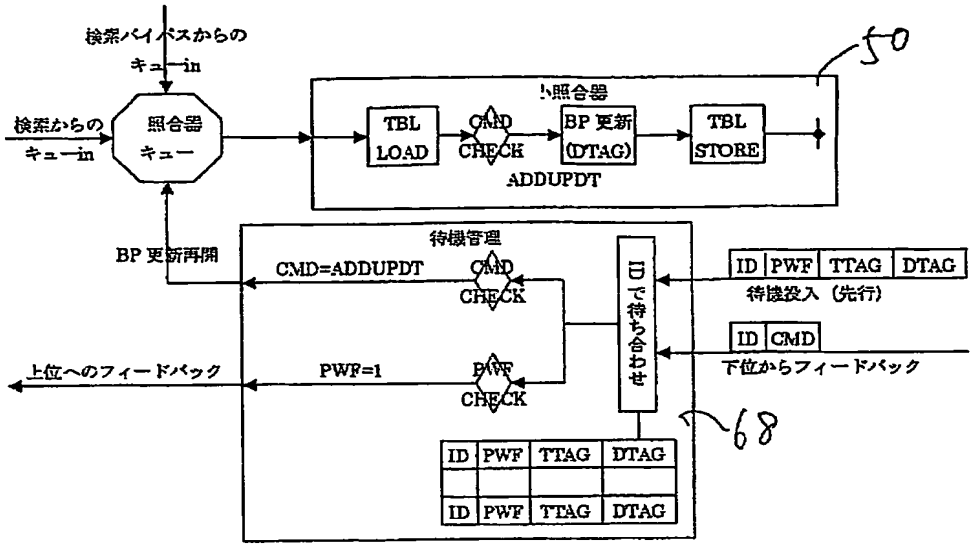
【図３９】


39
15
(検索バイパス用パス)
Q
照合器
50
Q
照合器
50
68
待機
管理
待機in
上位P.B.
68
待機
管理
待機in
上位P.B.

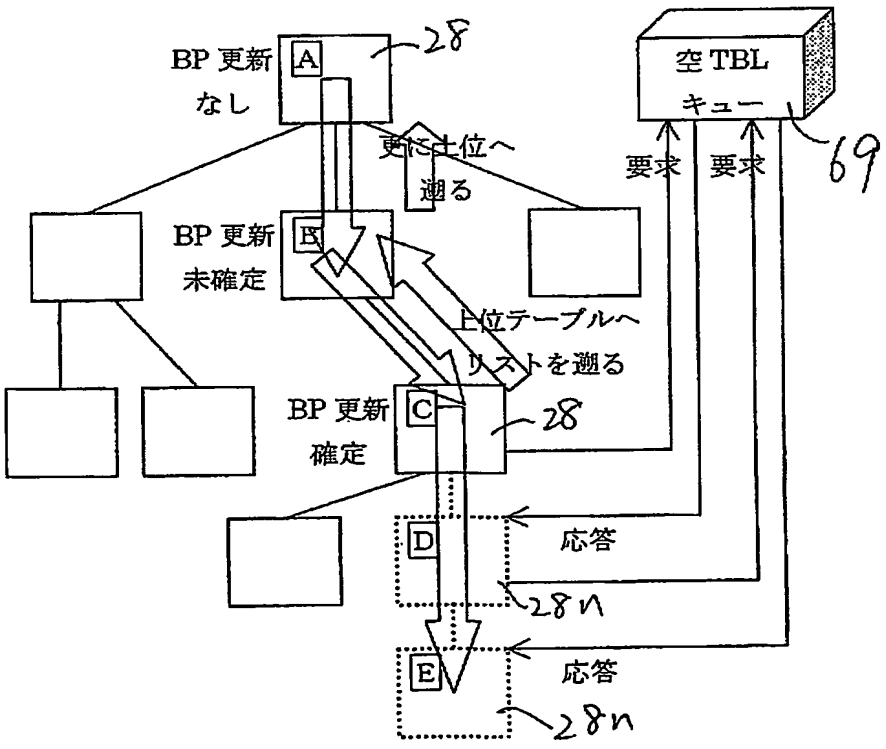
【図４０】


50
60
130
空き TBL 管理キュー
69
ADDNEW
MASK INIT
124
TBL 要求
123
126
127
ID
DTAG
KEY
CMD
TTAG
PWF
TBL LOAD
CMD CHECK
MASK READ
MASK CHECK
新規
MASK UPDATE WRITE
TAG WRITE
TTAGout
出力パケット合成(KEYはシフト・IDとDTAGは不変)
CMDout= ADDNEW
125
TAG READ
既存
128
CMDout= ADD
CMDout
129
ADD
ADDUPDT
更新確定
BP 更新 (DTAG)
PWFout =0
133
BYPASS CHECK
更新不要
131
132
更新保留
PWFout =1
PWFout
134
137
BP 更新 (DTAG)
PWF 0
CHECK
1
CMD=AD DWCLR
CMD=AD DUPDT
待機出力
136
135
BP 更新再開
待機管理
ID PWF TTAG DTAG
(PWF=1)
上位へのフィードバック
待ち合わせ
下位からの BP 更新指示フィードバック
ID CMD
上位への BP 更新指示フィードバック
68

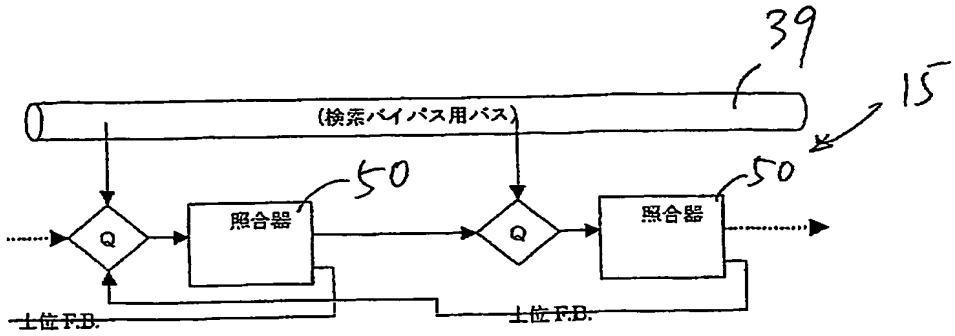
【図４１】


検索バイパスからの
キューin
検索からの
キューin
照合器
キュー
照合器
TBL
LOAD
CMD
CHECK
BP 更新
(DTAG)
TBL
STORE
ADDUPDT
50
待機管理
BP 更新再開
CMD=ADDUPDT
CMD
CHECK
ID で待ち合わせ
ID PWF TTAG DTAG
待機投入（先行）
ID CMD
下位からフィードバック
上位へのフィードバック
PWF=1
PWF
CHECK
68
ID PWF TTAG DTAG
ID PWF TTAG DTAG

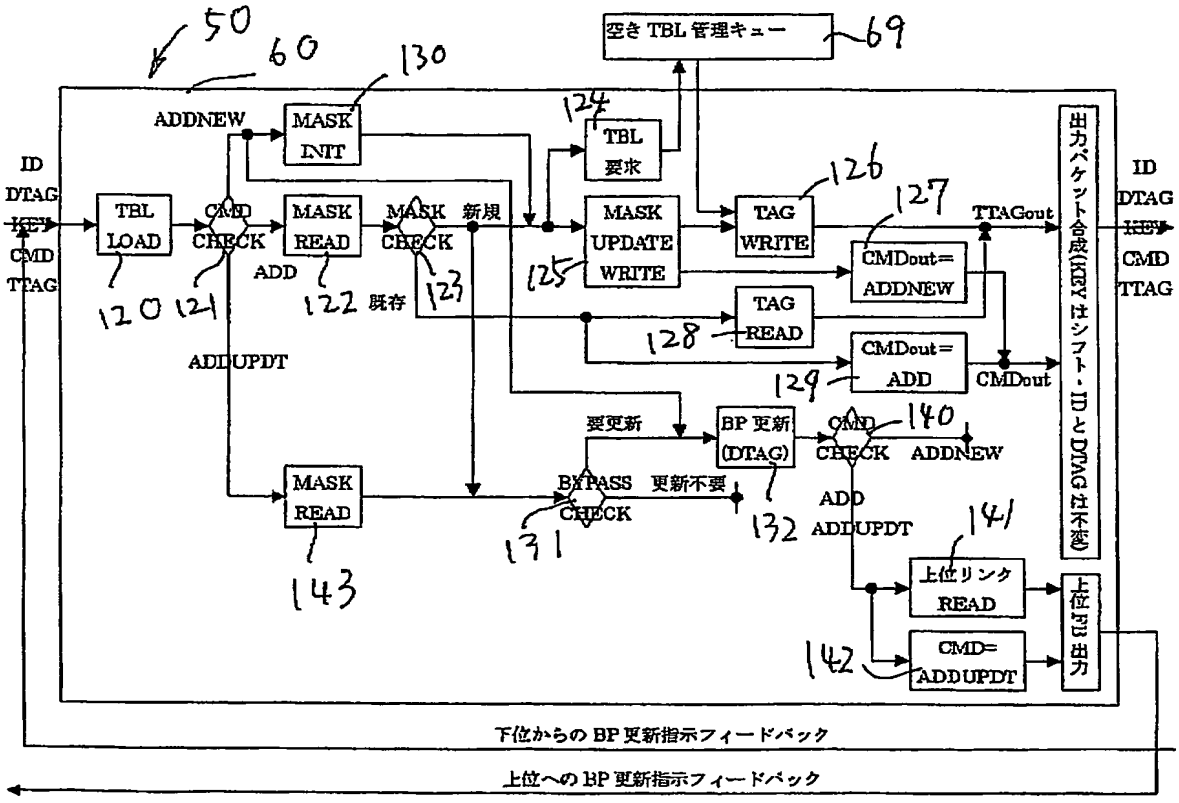
【図４２】


BP 更新
なし
A
28
空 TBL
キュー
69
更に上位へ
遡る
要求
要求
BP 更新
未確定
B
上位テーブルへ
リストを遡る
BP 更新
確定
C
28
D
応答
28n
E
応答
28n

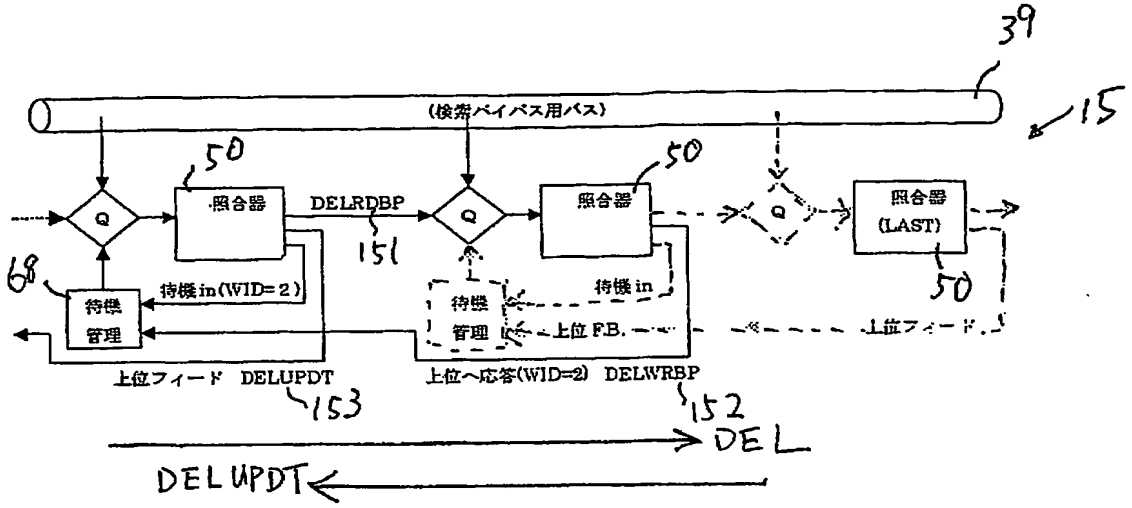
【図４３】


39
15
(検索バイパス用パス)
50
50
照合器
照合器
Q
Q
上位F.B.
上位F.B.

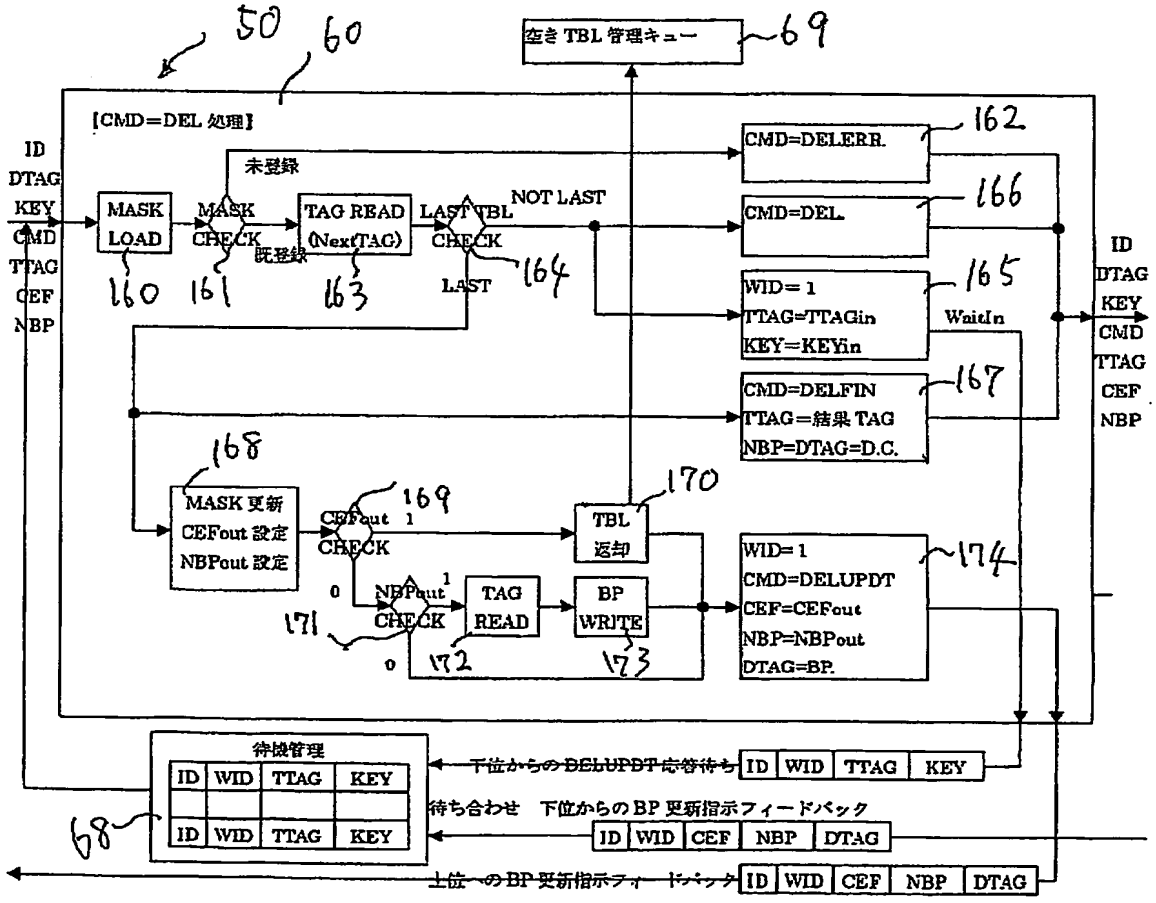
【図４４】


50
60
130
空き TBL 管理キュー
69
124
ADDNEW
MASK INIT
TBL 要求
ID
DTAG
KEY
CMD
TTAG
TBL LOAD
CMD CHECK
MASK READ
MASK CHECK
新規
ADD
既存
120
121
122
123
ADDUPDT
MASK UPDATE WRITE
125
TAG WRITE
126
127
TTAGout
CMDout= ADDNEW
TAG READ
128
CMDout= ADD
CMDout
129
出力パケット合成(KEYはシフト・IDとDTAGは不変)
ID
DTAG
KEY
CMD
TTAG
要更新
BP 更新 (DTAG)
CMD CHECK
ADDNEW
140
ADD
ADDUPDT
132
BYPASS CHECK
更新不要
131
MASK READ
143
上位リンク READ
141
CMD= ADDUPDT
142
上位FB出力
下位からの BP 更新指示フィードバック
上位への BP 更新指示フィードバック

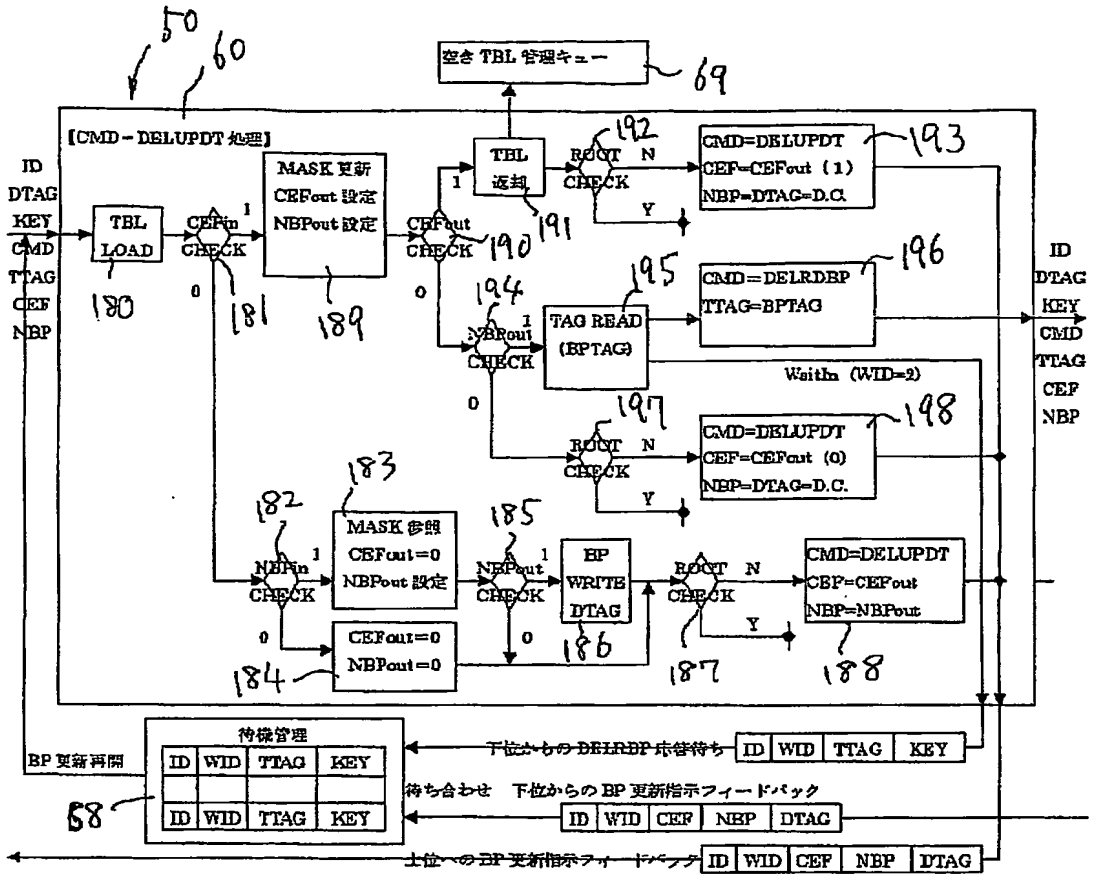
【図４５】


39
15
(検索バイパス用パス)
50
照合器
DELRDBP
151
照合器
50
照合器
(LAST)
50
68
待機
管理
待機in(WID=2)
待機
管理
待機in
上位F.B.
上位フィード
上位フィード DELUPDT
153
上位へ応答(WID=2) DELWRBP
152
DEL
DELUPDT

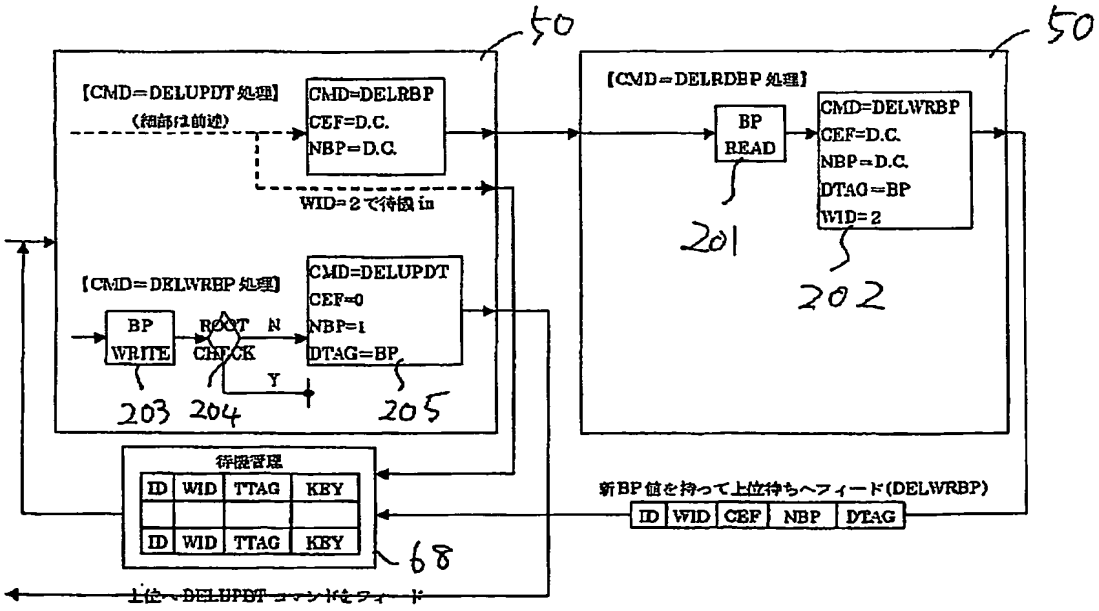
【図４６】


50
60
空き TBL 管理キュー
69
【CMD=DEL 処理】
ID
DTAG
KEY
CMD
TTAG
CEF
NBP
未登録
既登録
MASK LOAD
MASK CHECK
TAG READ (Next(TAG))
LAST TBL CHECK
NOT LAST
LAST
160
161
163
164
CMD=DELERR.
162
CMD=DEL.
166
WID=1
TTAG=TTAGin
KEY=KEYin
165
WaitIn
CMD=DELFIN
TTAG=結果 TAG
NBP=DTAG=D.C.
167
168
MASK 更新
CEFout 設定
NBPout 設定
CEFout CHECK
169
TBL 返却
170
NBPout CHECK
TAG READ
BP WRITE
171
172
173
WID=1
CMD=DELUPDT
CEF=CEFout
NBP=NBPout
DTAG=BP.
174
ID
DTAG
KEY
CMD
TTAG
CEF
NBP
待機管理
ID WID TTAG KEY
68
下位からの DELUPDT 応答待ち
ID WID TTAG KEY
待ち合わせ　下位からの BP 更新指示フィードバック
ID WID CEF NBP DTAG
上位への BP 更新指示フィードバック
ID WID CEF NBP DTAG

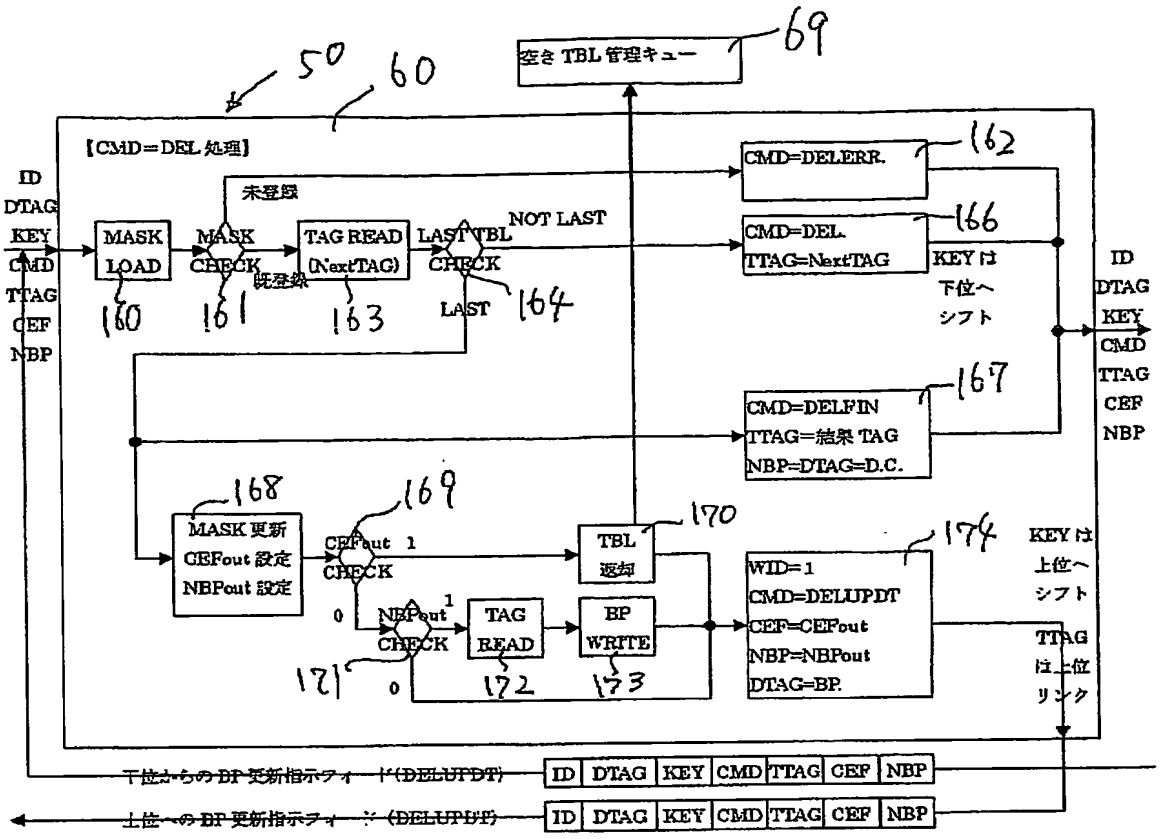
【図４７】


50
60
空き TBL 管理キュー
69
【CMD－DELUPDT 処理】
ID
DTAG
KEY
CMD
TTAG
CEF
NBP
TBL LOAD
CEFin CHECK
MASK 更新 CEFout 設定 NBPout 設定
TBL 返却
CEFout CHECK
ROOT CHECK
CMD=DELUPDT CEF=CEFout (1) NBP=DTAG=D.C.
CMD=DELRDBP TTAG=BPTAG
TAG READ (BPTAG)
NBPout CHECK
Within (WID=2)
CMD=DELUPDT CEF=CEFout (0) NBP=DTAG=D.C.
MASK 参照 CEFout=0 NBPout 設定
NBPin CHECK
NBPout CHECK
BP WRITE DTAG
CMD=DELUPDT CEF=CEFout NBP=NBPout
CEFout=0 NBPout=0
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
待機管理
ID WID TTAG KEY
68
BP 更新再開
下位からの DELRBP 応答待ち
待ち合わせ　下位からの BP 更新指示フィードバック
ID WID CEF NBP DTAG
上位への BP 更新指示フィードバック

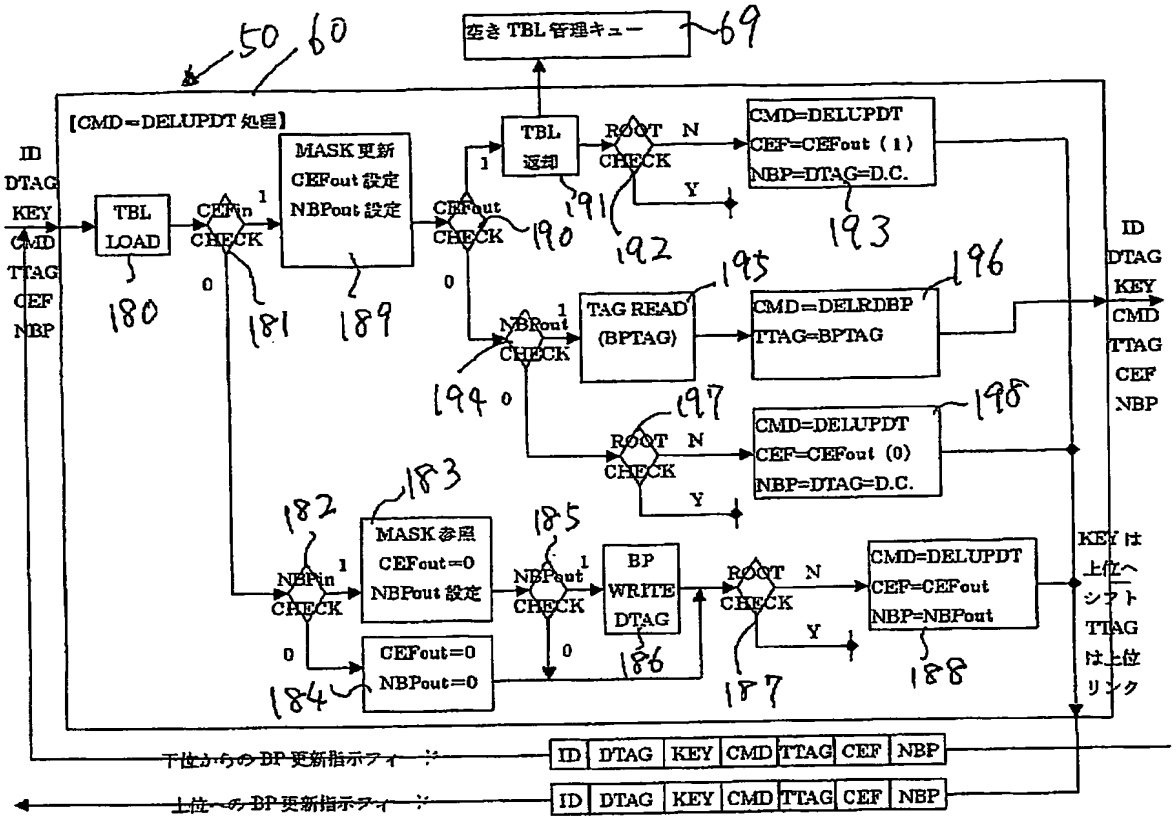
【図４８】


50
50
【CMD＝DELUPDT 処理】
（相部は前述）
CMD=DELRBP CEF=D.C. NBP=D.C.
WID=2で待機 in
【CMD＝DELRDBP 処理】
BP READ
CMD=DELWRBP CEF=D.C. NBP=D.C. DTAG=BP WID=2
201
202
【CMD＝DELWRBP 処理】
BP WRITE
ROOT CHECK
CMD=DELUPDT CEF=0 NBP=1 DTAG=BP
203
204
205
待機管理
ID WID TTAG KEY
68
新BP値を持って上位待ちへフィード(DELWRBP)
ID WID CEF NBP DTAG
上位へ DELUPDT コマンドをフィード

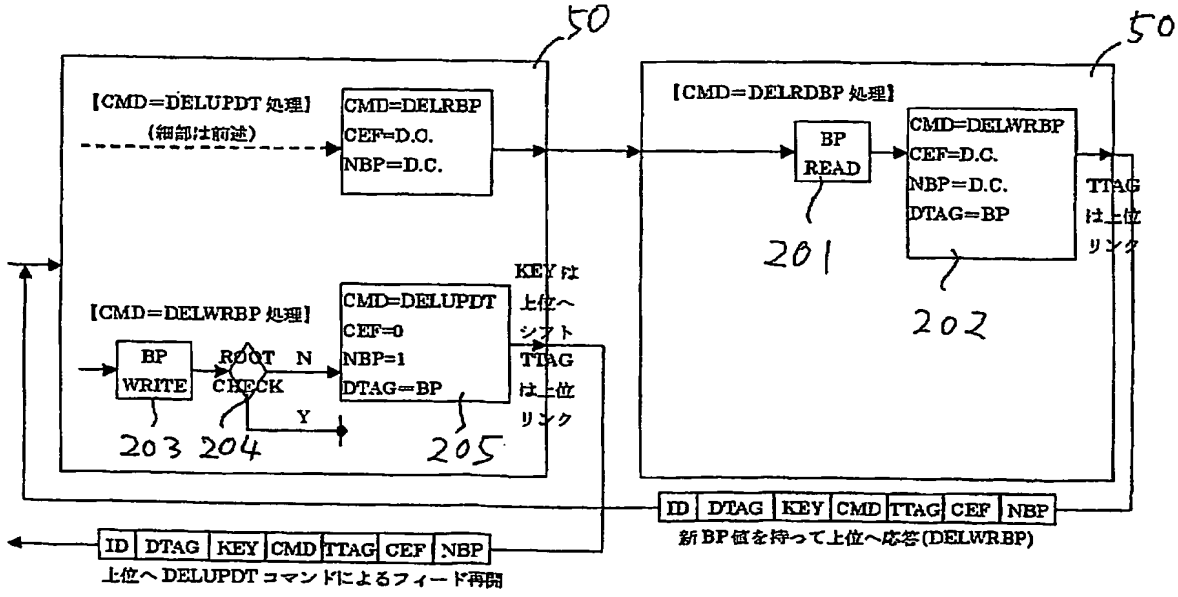
【図４９】


50
60
69
空き TBL 管理キュー
【CMD＝DEL 処理】
ID
DTAG
KEY
CMD
TTAG
CEF
NBP
MASK LOAD
MASK CHECK
未登録
既登録
TAG READ (NextTAG)
LAST TBL CHECK
NOT LAST
LAST
160
161
163
164
CMD=DELERR.
162
CMD=DEL.
TTAG=NextTAG
166
KEY は 下位へ シフト
CMD=DELFIN
TTAG=結果 TAG
NBP=DTAG=D.C.
167
168
MASK 更新
CEFout 設定
NBPout 設定
169
CEFout 1
CHECK
0
NBPout
1
CHECK
0
171
TAG READ
172
BP WRITE
173
TBL 返却
170
WID=1
CMD=DELUPDT
CEF=CEFout
NBP=NBPout
DTAG=BP.
174
KEY は 上位へ シフト
TTAG は上位リンク
下位からの BP 更新指示ﾌｫｰﾏｯﾄ (DELUPDT)
上位への BP 更新指示ﾌｫｰﾏｯﾄ (DELUPDT)
ID
DTAG
KEY
CMD
TTAG
CEF
NBP

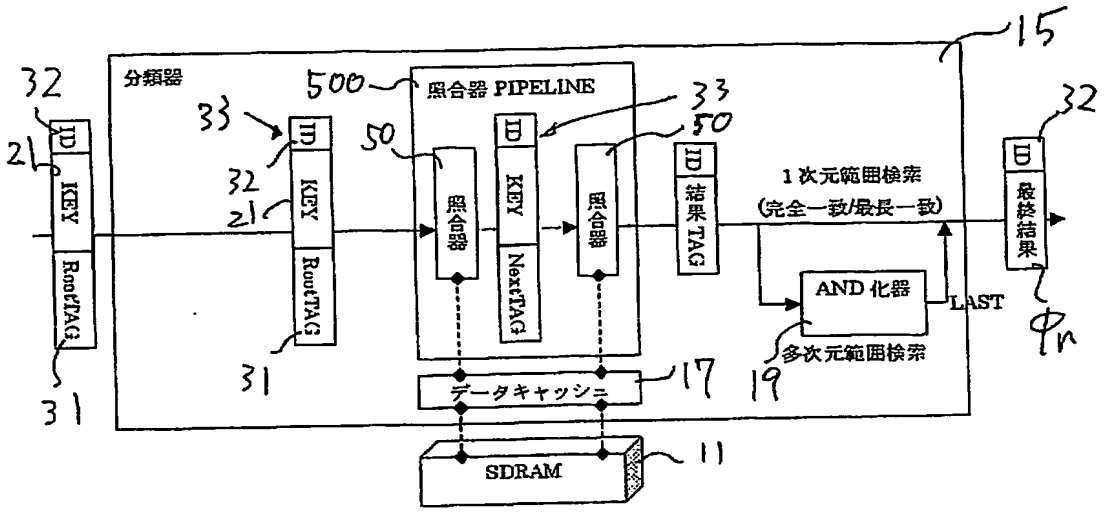
【図５０】


50
60
空きTBL管理キュー
69
【CMD=DELUPDT処理】
ID
DTAG
KEY
CMD
TTAG
CEF
NBP
TBL LOAD
180
CEFin CHECK
181
MASK更新
CEFout設定
NBPout設定
189
TBL返却
191
CEFout CHECK
190
ROOT CHECK
192
CMD=DELUPDT
CEF=CEFout（1）
NBP=DTAG=D.C.
193
TAG READ (BPTAG)
195
CMD=DELRDBP
TTAG=BPTAG
196
NBPout CHECK
194
CMD=DELUPDT
CEF=CEFout（0）
NBP=DTAG=D.C.
197
198
182
MASK参照
CEFout=0
NBPout設定
183
NBPin CHECK
184
CEFout=0
NBPout=0
185
NBPout CHECK
186
BP WRITE DTAG
ROOT CHECK
187
CMD=DELUPDT
CEF=CEFout
NBP=NBPout
188
KEYは上位へシフト
TTAGは上位リンク
下位からのBP更新指示フィード
上位へのBP更新指示フィード
ID DTAG KEY CMD TTAG CEF NBP

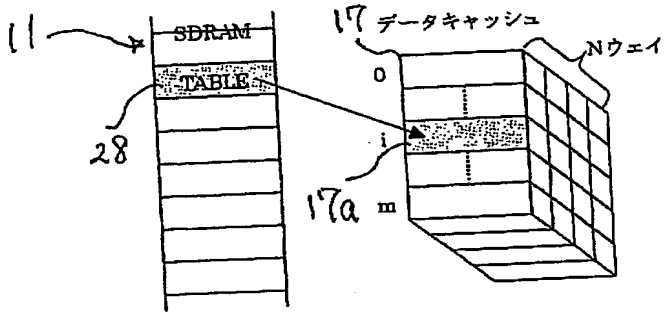
【図５１】


50
50
【CMD=DELUPDT処理】
（細部は前述）
CMD=DELRBP
CEF=D.C.
NBP=D.C.
【CMD=DELRDBP処理】
BP READ
201
CMD=DELWRBP
CEF=D.C.
NBP=D.C.
DTAG=BP
202
TTAGは上位リンク
【CMD=DELWRBP処理】
BP WRITE
203
ROOT CHECK
204
CMD=DELUPDT
CEF=0
NBP=1
DTAG=BP
205
KEYは上位へシフト
TTAGは上位リンク
ID DTAG KEY CMD TTAG CEF NBP
新BP値を持って上位へ応答(DELWRBP)
上位へDELUPDTコマンドによるフィード再開

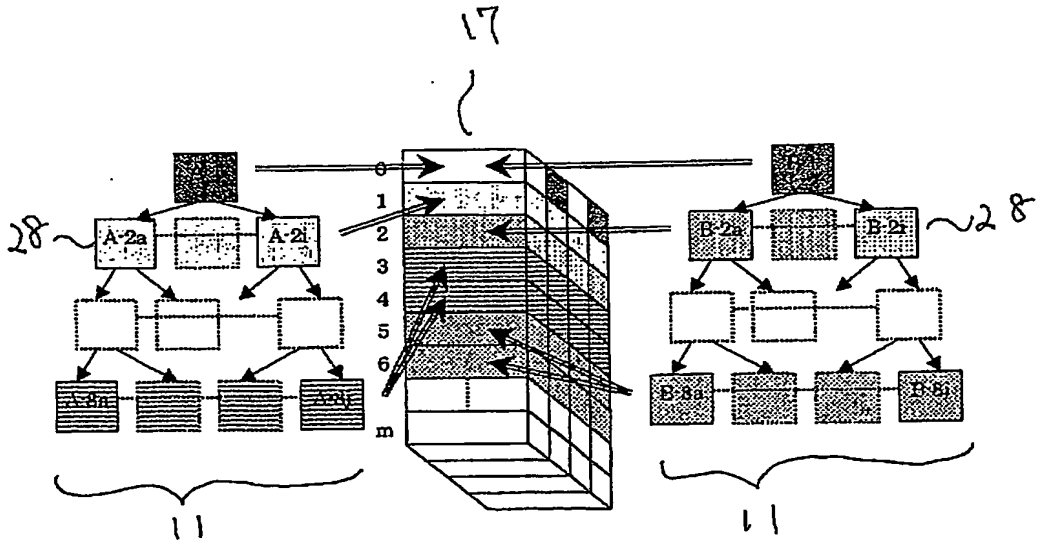
【図５２】


15
32
分類器
500
照合器 PIPELINE
33
33
50
50
32
ID
KEY
RootTAG
ID
KEY
RootTAG
ID
KEY
NextTAG
照合器
照合器
ID
結果 TAG
1次元範囲検索
(完全一致/最長一致)
ID
最終結果
32
AND 化器
LAST
多次元範囲検索
21
21
31
31
データキャッシュ
17
19
SDRAM
11

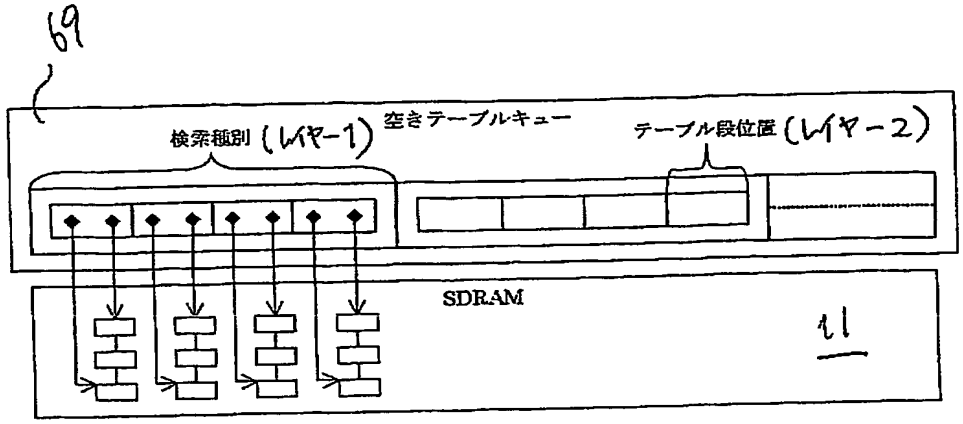
【図５３】


11
SDRAM
TABLE
28
17 データキャッシュ
Nウェイ
0
i
17a
m

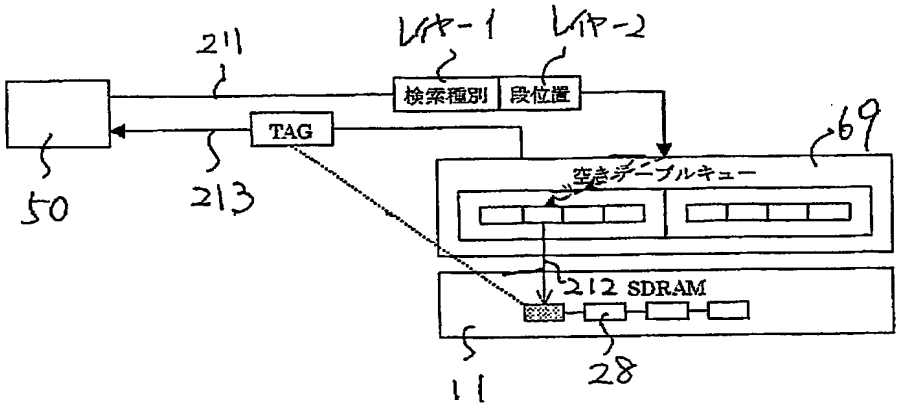
【図５４】


17
28
A-2a
A-2i
A-3m
A-3j
0
1
2
3
4
5
6
m
B-2a
B-2i
28
B-3a
B-3i
11
11

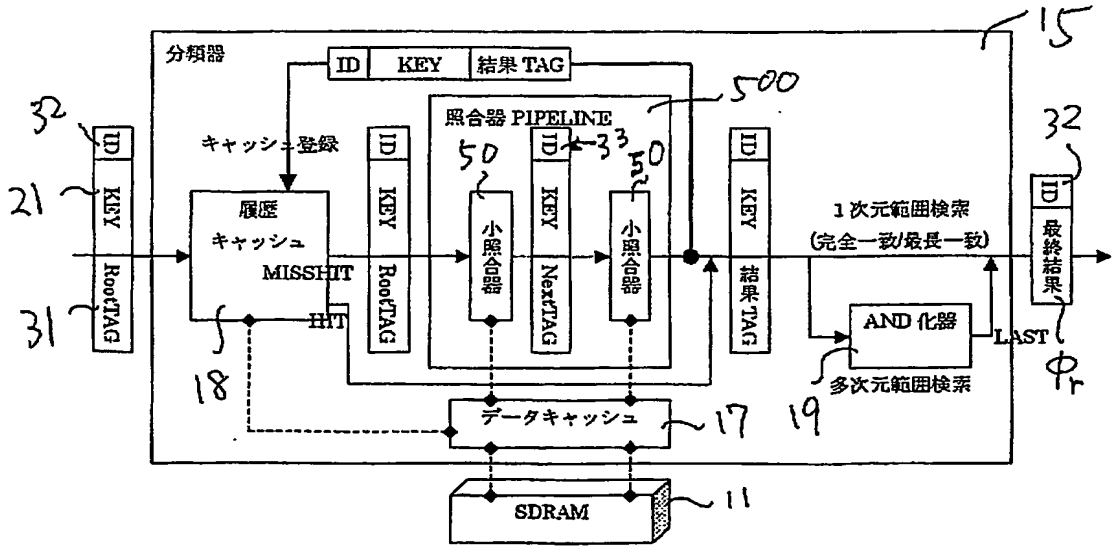
【図５５】


69
検索種別（レイヤ-1）
空きテーブルキュー
テーブル段位置（レイヤ-2）
SDRAM
11

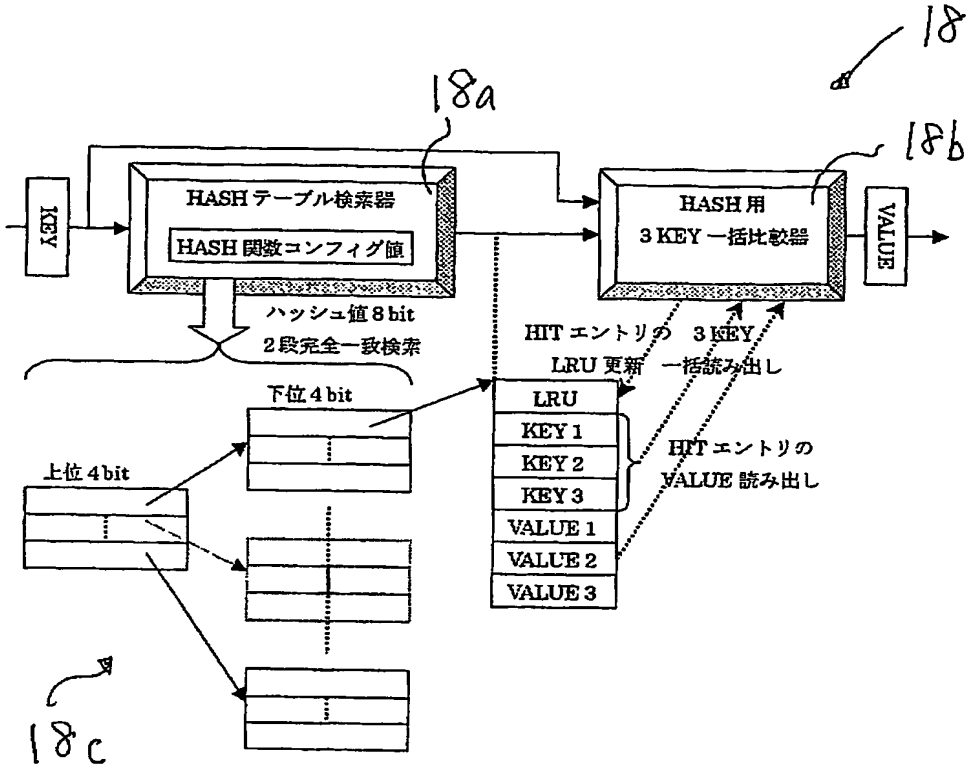
【図５６】


211
レイヤ-1
レイヤ-2
検索種別
段位置
TAG
69
空きテーブルキュー
212 SDRAM
50
213
11
28

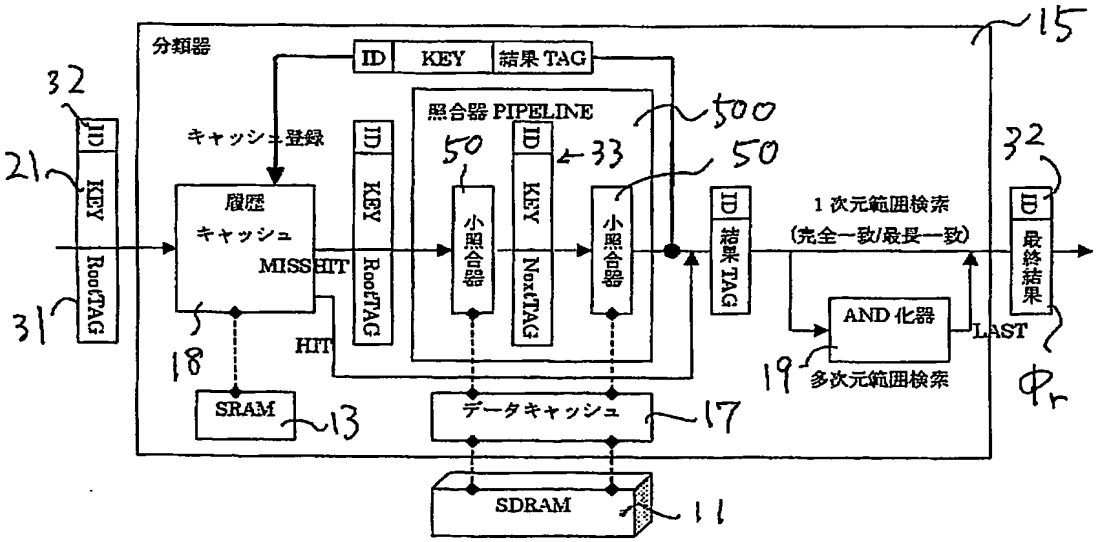
【図５７】


15
分類器
ID
KEY
結果 TAG
500
照合器 PIPELINE
32
キャッシュ登録
50
33
52
1次元範囲検索
(完全一致/最長一致)
最終結果
21
履歴
キャッシュ
MISSHIT
HIT
RootTAG
NextTAG
小照合器
AND 化器
LAST
多次元範囲検索
31
18
19
データキャッシュ
17
SDRAM
11

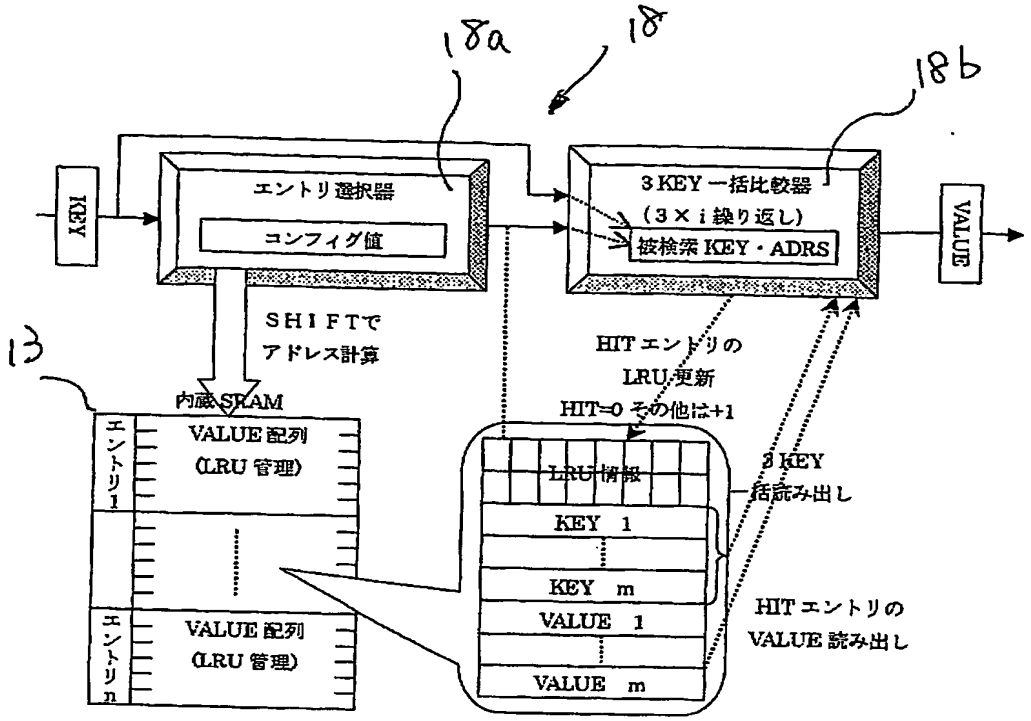
【図５８】


18
18a
18b
KEY
HASH テーブル検索器
HASH 関数コンフィグ値
HASH 用
3 KEY 一括比較器
VALUE
ハッシュ値 8 bit
2 段完全一致検索
HIT エントリの 3 KEY
LRU 更新 一括読み出し
下位 4 bit
上位 4 bit
LRU
KEY 1
KEY 2
KEY 3
VALUE 1
VALUE 2
VALUE 3
HIT エントリの
VALUE 読み出し
18c

【図５９】


15
分類器
ID
KEY
結果 TAG
32
21
31
キャッシュ登録
照合器 PIPELINE
500
50
33
履歴
キャッシュ
MISS
HIT
RootTAG
NextTAG
小照合器
結果 TAG
1 次元範囲検索
(完全一致/最長一致)
最終結果
AND 化器
LAST
19
多次元範囲検索
18
SRAM
13
データキャッシュ
17
SDRAM
11
Φr

【図６０】



【図６１】

$$Ai \times Bj = \sum_{k} aik * bkj$$

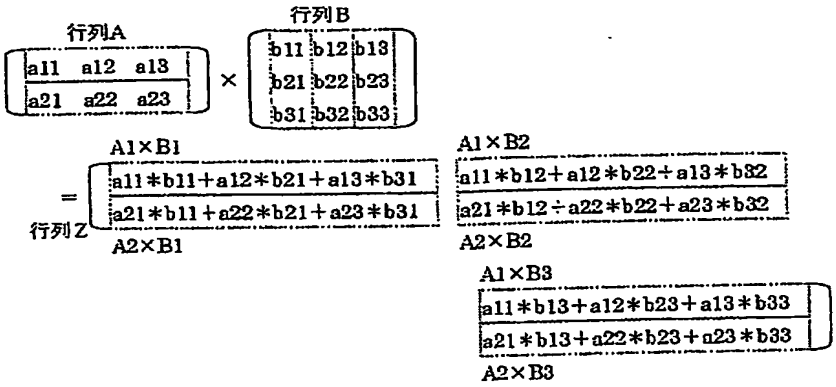
- Σ内はブール乗算（＊）とブール加算（＋）に展開される。
- ここでのブール乗算・ブール加算の定義は、ブール乗算「＊」はENOR（Exclusive NOR、非排他論理和）、ブール加算「＋」はANDである

（ブール加算）１＋１＝１　１＋０＝０　０＋１＝０　０＋０＝０

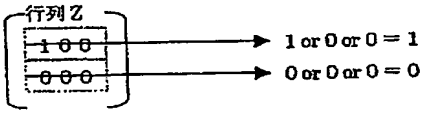
（ブール乗算）１＊１＝１　１＊０＝０　０＊１＝０　０＊０＝１

- ENORは比較回路の＝のロジックと同じであるので、Ai＊Bjは比較等号回路をANDで結んだものだと理解してもかまわない。行列で理解したほうが見通しが良い。

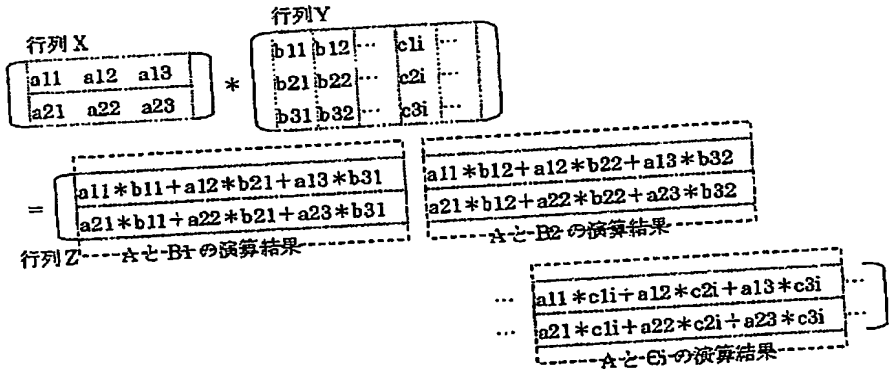
【図６２】


行列A
a11 a12 a13
a21 a22 a23
×
行列B
b11 b12 b13
b21 b22 b23
b31 b32 b33
=
行列Z
A1×B1
a11*b11+a12*b21+a13*b31
a21*b11+a22*b21+a23*b31
A2×B1
A1×B2
a11*b12+a12*b22+a13*b32
a21*b12+a22*b22+a23*b32
A2×B2
A1×B3
a11*b13+a12*b23+a13*b33
a21*b13+a22*b23+a23*b33
A2×B3

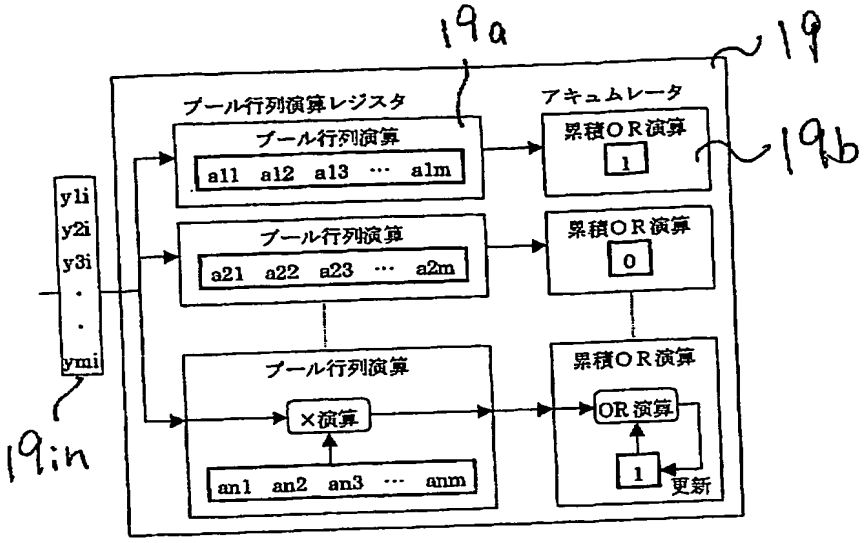
【図６３】


行列Z
1 0 0
0 0 0
1 or 0 or 0 = 1
0 or 0 or 0 = 0

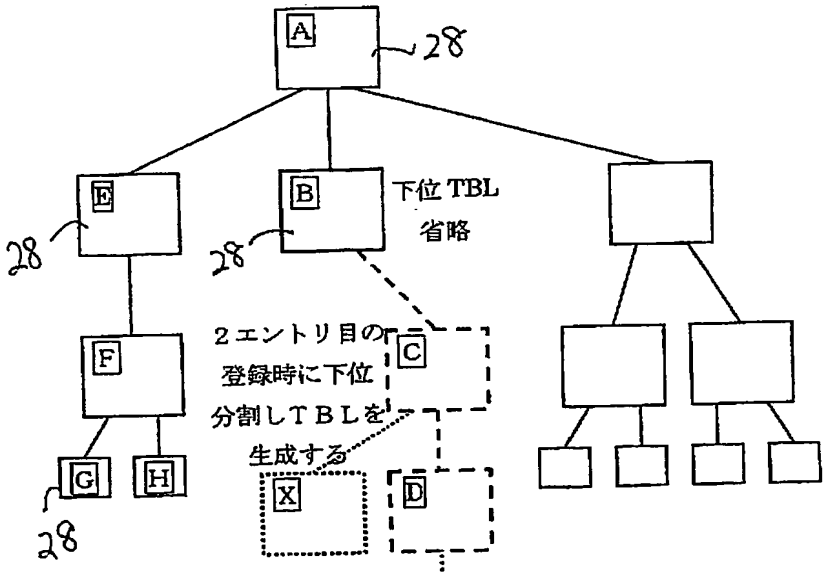
【図６４】


行列X
a11 a12 a13
a21 a22 a23
*
行列Y
b11 b12 … c1i …
b21 b22 … c2i …
b31 b32 … c3i …
=
行列Z
a11*b11+a12*b21+a13*b31
a21*b11+a22*b21+a23*b31
Aとb1の演算結果
a11*b12+a12*b22+a13*b32
a21*b12+a22*b22+a23*b32
Aとb2の演算結果
… a11*c1i+a12*c2i+a13*c3i …
… a21*c1i+a22*c2i+a23*c3i …
Aとciの演算結果

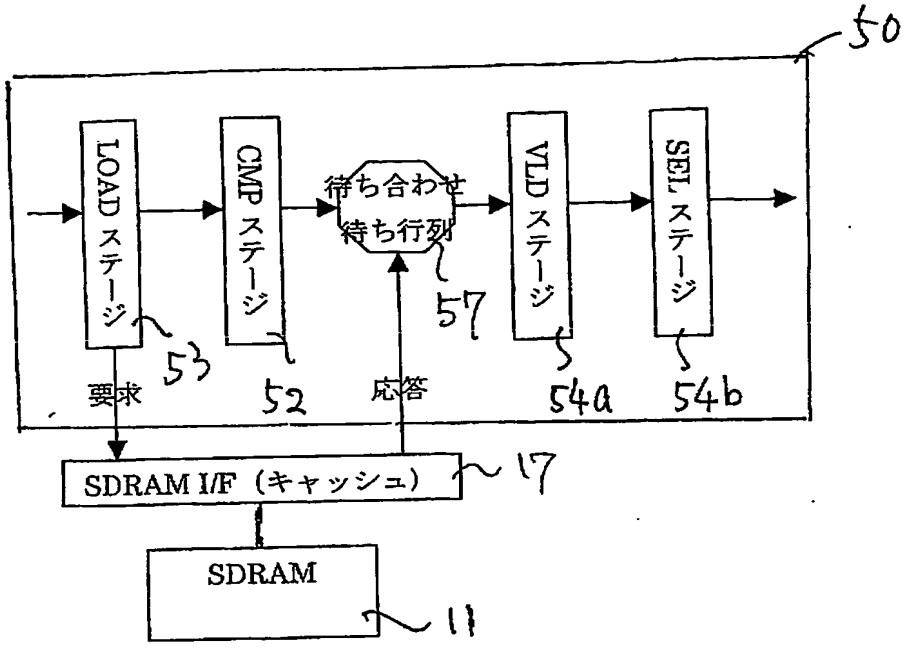
【図６５】


19a
19
プール行列演算レジスタ
アキュムレータ
プール行列演算
a11 a12 a13 … a1m
累積ＯＲ演算
1
19b
y1i
y2i
y3i
ymi
プール行列演算
a21 a22 a23 … a2m
累積ＯＲ演算
0
19in
プール行列演算
×演算
an1 an2 an3 … anm
累積ＯＲ演算
OR演算
1
更新

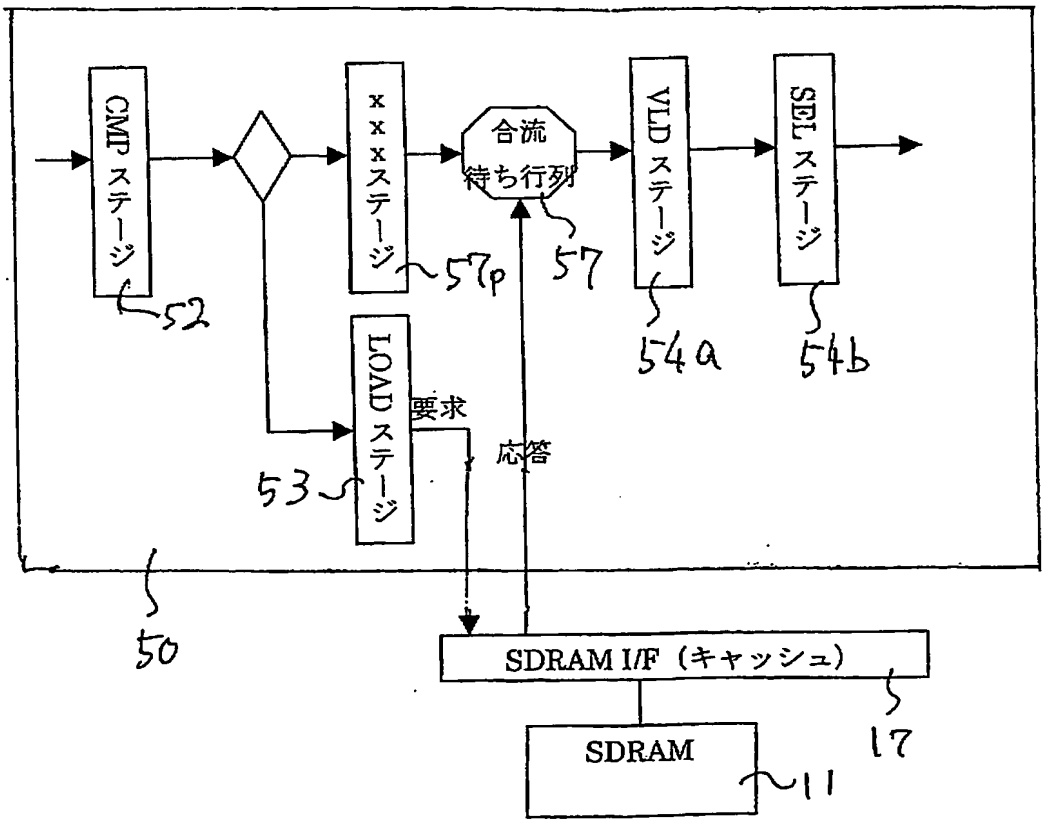
【図６６】


A
28
E
B
下位 TBL
省略
28
28
F
２エントリ目の
登録時に下位
分割しＴＢＬを
生成する
C
G
H
X
D
28

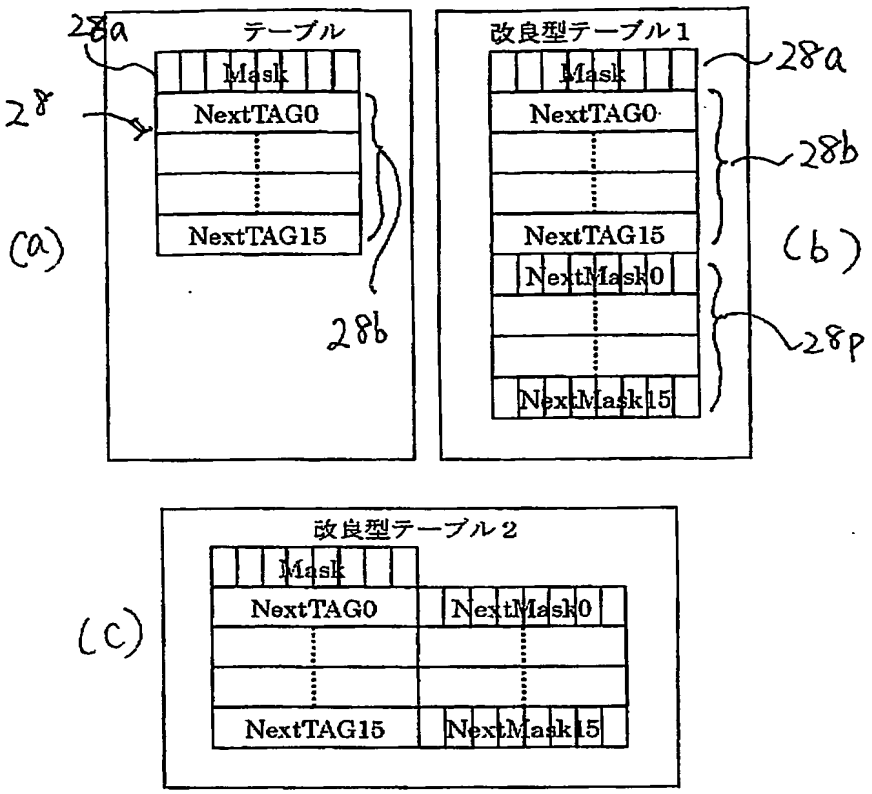
【図６７】


50
LOADステージ
CMPステージ
待ち合わせ
待ち行列
VLDステージ
SELステージ
53
52
57
54a
54b
要求
応答
SDRAM I/F（キャッシュ）
17
SDRAM
11

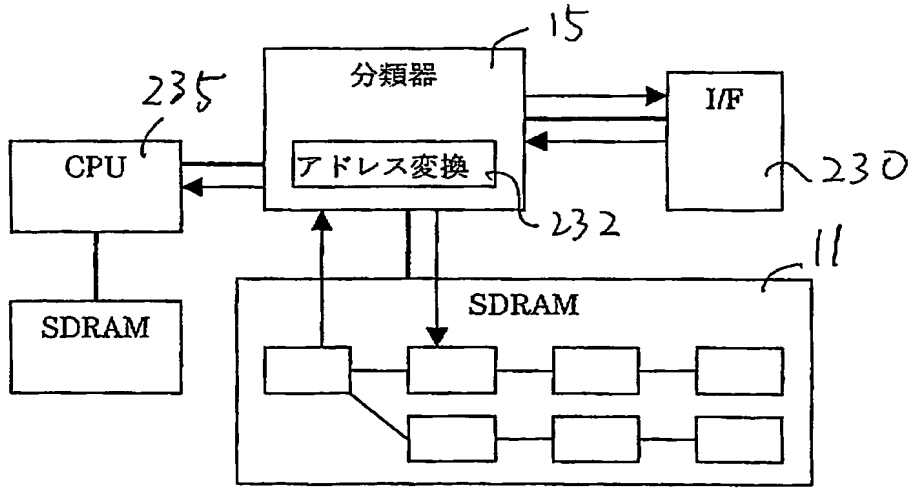
【図６８】


CMPステージ
xxxステージ
合流
待ち行列
VLDステージ
SELステージ
52
57p
57
54a
54b
LOADステージ
要求
応答
53
50
SDRAM I/F（キャッシュ）
17
SDRAM
11

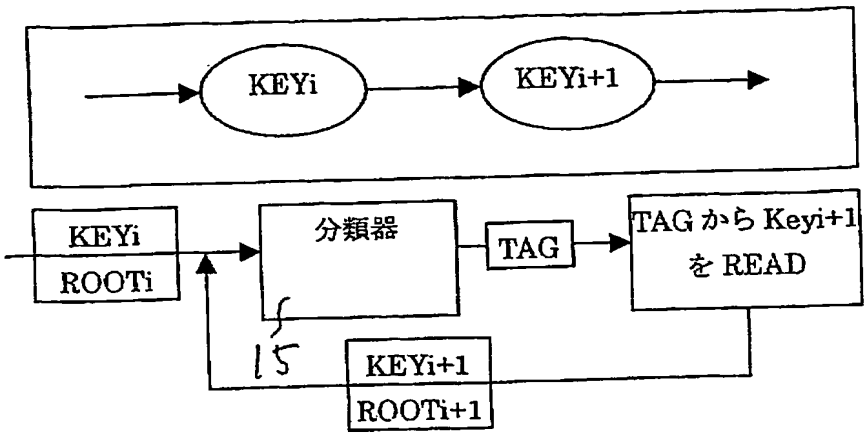
【図６９】


テーブル
28a
28
Mask
NextTAG0
NextTAG15
(a)
28b
改良型テーブル１
Mask
NextTAG0
NextTAG15
NextMask0
NextMask15
28a
28b
(b)
28p
改良型テーブル２
Mask
NextTAG0
NextMask0
NextTAG15
NextMask15
(c)

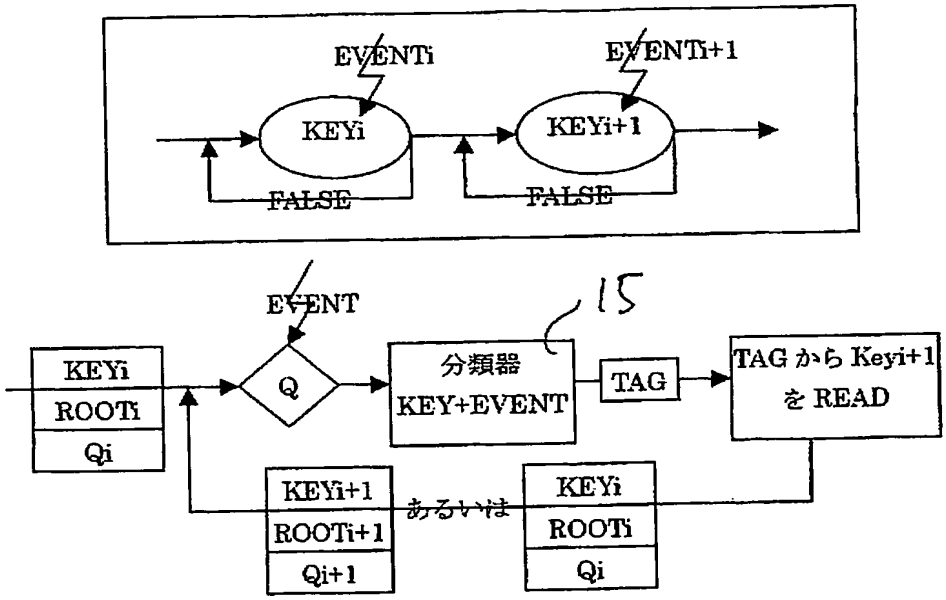
【図７０】


15
分類器
235
CPU
アドレス変換
232
I/F
230
11
SDRAM
SDRAM

【図７１】


KEYi
KEYi+1
KEYi
ROOTi
分類器
TAG
TAG から Keyi+1
を READ
15
KEYi+1
ROOTi+1

【図７２】


EVENTi
EVENTi+1
KEYi
KEYi+1
FALSE
FALSE
EVENT
15
KEYi
ROOTi
Qi
Q
分類器
KEY+EVENT
TAG
TAG から Keyi+1
を READ
KEYi+1
ROOTi+1
Qi+1
あるいは
KEYi
ROOTi
Qi

【書類名】　要約書

【要約】

【課題】　範囲一致を始め、完全一致および最長一致といった検索機能を提供可能であり、上位アプリケーションソフトウェアの負担が軽いビットストリングの照合装置を提供する。

【解決手段】　照合装置５０においては、部分被検索キー３６を比較ユニット５２により取り得る全ての値と照合し、その照合段階のエントリを示すマスクデータ２８ａを用いて判定ユニット５４により部分被検索キー３６の照合結果を得る。この照合装置５０においては、照合テーブル２８にエントリを追加削除する際にエントリ移動がなく、上位アプリケーションによるテーブル管理のオーバヘッドを縮小または排除できる。

【選択図】　図１８

認定・付加情報

特許出願の番号	特願２００２－２９１７０８
受付番号	５０２０１４９３４９２
書類名	特許願
担当官	第八担当上席　　００９７
作成日	平成１４年１０月　４日

＜認定情報・付加情報＞

【提出日】　平成14年10月 3日

次頁無

【書類名】　出願人名義変更届（一般承継）
【整理番号】　020320P531
【あて先】　特許庁長官殿
【事件の表示】
　【出願番号】　特願2002-291708
【承継人】
　【識別番号】　503253828
　【氏名又は名称】　株式会社インフォーエス
【承継人代理人】
　【識別番号】　100102934
　【弁理士】
　【氏名又は名称】　今井　彰
【提出物件の目録】
　【包括委任状番号】　0311781

認定・付加情報

特許出願の番号	特願２００２－２９１７０８
受付番号	５０３０１６２９３４０
書類名	出願人名義変更届（一般承継）
担当官	塩野　実　　２１５１
作成日	平成１５年１１月１８日

＜認定情報・付加情報＞

【提出日】　平成15年10月 1日

特願２００２－２９１７０８

出　願　人　履　歴　情　報

識別番号　　［５０２２５４９８１］

１．変更年月日　　２００２年　７月１５日
［変更理由］　　新規登録
住　所　　東京都中央区日本橋浜町３－１６－７
氏　名　　有限会社ディシーエル

特願２００２－２９１７０８

出　願　人　履　歴　情　報

識別番号　　［５０３２５３８２８］

１．変更年月日　　２００３年　７月１５日
［変更理由］　　新規登録
住　所　　東京都中央区日本橋浜町三丁目１６番７号
氏　名　　株式会社インフォーエス